For Reference

NOT TO BE TAKEN FROM THIS ROOM

Ex libris
universitatis
albertaensis

QUAECUMQUE VERA

The University of Alberta
Printing Department
Edmonton, Alberta

Digitized by the Internet Archive
in 2023 with funding from
University of Alberta Library

https://archive.org/details/Lo1972

THE UNIVERSITY OF ALBERTA

AN AUTOMATIC OPTIMUM ITERATIVE FEEDBACK DOCUMENT RETRIEVAL SYSTEM

by

© Adrian K. Lo

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTING SCIENCE

EDMONTON, ALBERTA

FALL, 1972

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled AN AUTOMATIC OPTIMUM ITERATIVE FEEDBACK DOCUMENT RETRIEVAL SYSTEM, submitted by Adrian K. Lo in partial fulfillment of the requirements for the degree of Master of Science.

ABSTRACT

The basic design problems of a document retrieval system are reviewed. A simple optimum iterative feedback system is proposed that makes use of two interrelated sets of parameters supplied respectively by the user and the system. The set of user parameters is designed to reflect the user's own point of view on the search subject matter; while the set of system parameters is designed to reveal some data base characteristics. A set of index terms and their corresponding significance values are abstracted from the Computing Science data base by an automatic indexing algorithm based on some statistical association measures. In order to eliminate storage shortage problems created by large matrices such as the document-term matrix, a least-storage scheme and a subscript-matching algorithm are developed to assist manipulations of these large matrices. Some relevance judgment criteria are defined and a relevance measure is derived. The optimum iterative feedback algorithm is first described for search on document title terms only; and is then generalized to include search on other relevant items such as author names and so on. Finally, the convergence of the algorithm is verified for both cases.

ACKNOWLEDGEMENTS

I wish to express my sincere appreciation to Professor H.S. Heaps for his constant advice, guidance, encouragement, and financial support at all stages of the research. I am also indebted to Dr. K.V. Leung for many of his valuable comments and suggestions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 The Basic Problems

The design of an automatic information retrieval system includes basically the organization of data; the manipulation of information; the formulation of search logic; the definition of query language; the implementation of search strategy; the presentation of output; the evaluation of system performance; and, finally, the optimization of system effectiveness. Since information retrieval systems are user-oriented in nature, the prime objective of system design is to achieve the retrieval of all relevant, and only relevant, information in response to any user's query. An ideal system can thus be regarded as one that retrieves from a given data base all the relevant information while at the same time rejecting all information that is irrelevant to any given search request. However, such an ideal system never occurs in practice.

Indeed, there are numerous factors that govern retrieval performance. Human errors and system incompatibilities are the major sources of discrepancy. Human errors may be further subdivided into designer errors and user errors. Examples of designer errors are inaccurate representation of information, such as through spelling errors; poor search strategy; and ambiguities in formal query language definition. Common user errors arise from poor request formulation and poor concepts of

system capabilities. Examples of system incompatibilities are bugs in programming; poor decision-making such as in determining a threshold value at a certain stage of the search process; and inconsistency between search logic and search functions resulting in misinterpretation of query. Fortunately, these deficiencies are normally controllable by means of careful planning and management. In many systems, some form of optimization process may be employed in order to reduce the noise in the search output so as to ensure satisfactory responses to the search requests.

This leads to the remaining, and yet the most controversial problem, which is the judgment of relevance of the final search output. Obviously, relevance assessment is totally subjective to the individual's viewpoint. The user, for instance, is primarily interested in obtaining information that satisfies his particular need; and not in whether the retrieved information does, in fact, match his search request. The system designer, on the other hand, has to make sure that the retrieved information matches the logic of the query. Thus, relevance judgment in this context is rather unreliable. An alternative is to employ a few judges or groups of judges at one time. However, experimental results indicate that under different conditions even the same judge may give different relevance judgment to the same query and the same corresponding set of output. It is not until some specific guidelines are followed that a substantial gain of stability in relevance judgment is observed [1,2]. Consequently, it may be hypothesized that a set of well-defined relevance

judgment criteria is absolutely essential for any retrieval system whose performance depends heavily on some relevance measures. Thence, the optimization of search output can be carried out without ambiguity.

1.2 Optimization Methods

Most retrieval systems that optimize search output make use of iterative search techniques. A fairly straightforward semiautomatic approach is to present the user with the search output together with a set of machine-generated index terms derived on the basis of their association within the system. The user then makes a relevance judgment according to some predefined criteria. If the search output is not satisfactory, he may reformulate his initial query by selecting more significant terms from the index list and resubmit a modified query for another search run. It is expected that the revised query will lead to better search results because it reflects the system parameters as well as the user's own point of view. He may repeat the same precedures until a fully satisfactory output is obtained [3]. However, this method of search optimization is rather inefficient in that it is too time-consuming and costly. In some cases, the users may soon become very frustrated in the process of waiting and repeating the same routine over and over again.

A more sophisticated approach that has been widely experimented with is the use of real-time, man-machine interaction [4 to 6]. The basic principles behind this method

are exactly the same as in the one just described. The system makes a finite number of searches interspersed with user reformulation of search request with the aid of additional index terms displayed on the terminal. The user may find terminal use to be amusing for the first few times. After a while he will probably become bored at having to wait for response and make changes. As a result of all the inconvenience he may sometimes lose track of his original information. Furthermore, as most computer systems give terminal jobs the highest priority for execution, the cost for on-line iterative searches is far more expensive than for other searches. Finally, the most serious drawback of on-line searches is that terminals are then often unnecessarily denied for use for other purposes. From these observations, it can be concluded that a more economical and effective system is to be preferred.

In a recent study by Heaps and Ko [7 to 9] a method known as the "automatic adaptive processing of questions" is examined. Four criteria are derived and tested separately. The users need only specify an estimate of relevance to their search requests. The system then modifies the requests automatically according to one of the four criteria and obtains an optimum set of weights for internal use. Search results show that the final output may contain some relevant information that the requesters neglected to mention in their queries. The non-iterative and completely automatic nature of this model has successfully eliminated the painstaking and time-consuming efforts normally required by the users of other systems. However, the system is not without

shortcomings. The main one is the requirement of more computer time because large matrices, and a lot of computations, are involved.

1.3 The Model

An alternative approach can be represented by a simple model as shown in Fig. 1.1. The model may be called an automatic optimum iterative feedback document retrieval system because it makes use of automatic iterative feedback control to optimize search outputs. The method consists of three phases, namely, the pre-search phase, the search-phase, and the post-search phase. In the pre-search phase, the user formulates his search requests with the aid of a set of index terms and their significance values automatically abstracted from the data base according to some attribute measures based on statistical associations. He then submits his requests coded in conformity with the query language described in Backus Normal Form. In this manner, ambiguous requests can readily be detected and then rejected. As a result, the acceptable requests will be assumed to contain all relevant information needed for search purposes as well as search optimization purposes.

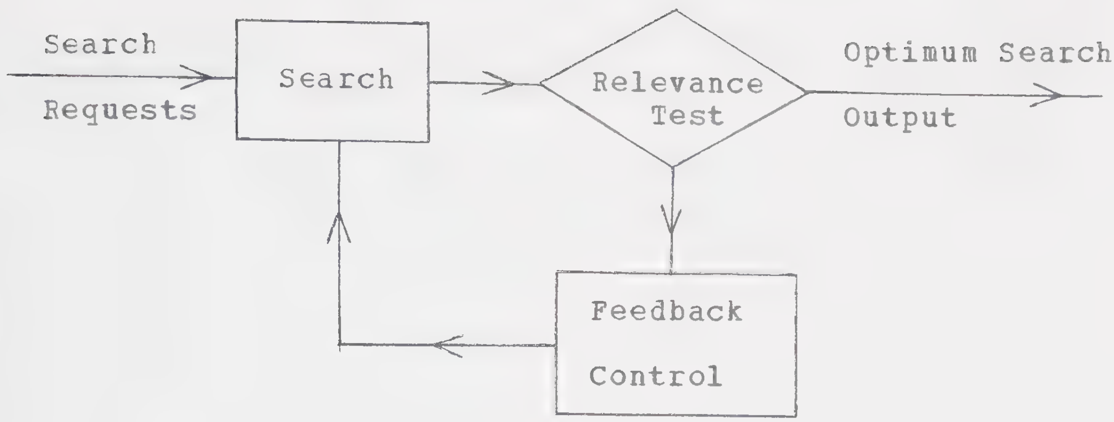


Fig. 1.1 The Model

The search phase is responsible for deciding the degree of relevance that a document has in relation to the requests. A document is classed as provisionally relevant only if its relevance value is greater than, or equal to, a pre-determined cutoff value. Members of the set of provisionally relevant documents, if not null, will be arranged in descending order of relevance. The post-search phase then examines this set to determine if some pre-defined relevance criteria are met. Whenever it is not met, control is passed back to the search phase after modifying the initial (or previously modified) search requests. Such examination and modification is repeated a finite number of times until the relevance criteria are met.

CHAPTER II

AN AUTOMATIC INDEXING ALGORITHM

2.1 The Data Base

The data base used is the CSDATA tape prepared for experimental use for Course CS560 within the Department of Computing Science at the University of Alberta. It is made up of approximately 7,000 journal articles taken from various current computing science journals. Each journal name is represented by an ASTM (American Society for Testing and Materials) coden. A list of ASTM codens and journal names used in CSDATA is included in Appendix A.

To facilitate editing and updating of data, the tape is blocked into logical records of 80 bytes according to the format as shown in Fig. 2.1. All author names (excluding initials) and title words are truncated to five letters of each. The former is followed by a slash (/), and the latter by a blank. Words of less than five letters are left-justified and followed by the appropriate number of blanks. Hyphenated title words are coded as separate words, while all insignificant words such as prepositions are eliminated. In the case when more than one logical record is required, the letter 'C' is specified in column 80 to indicate continuation of data to columns 14-79 of the next logical record. The data in columns 1-13 are repeated for the purpose of article identification. Finally, the data base is sorted alphanumerically in ascending order on columns 1-

13, and with a secondary sort in descending order on column 80.

1 5	6 7	8 9	10 13	14 79	80
Coden	Year	Volume	Starting Page Number	{Author Name}, {Title Word}	Cont.

Fig. 2.1 <u>Format of a Logical Record</u>

The following is a typical example of a logical record belong to the data base:

JACOA66130317KRISH/WOOD /TIME SHARE OPERA INTER SERVI TIMES EXPON

which is the code for an article appearing in Vol. 13, 1966, of the Journal of the Association for Computing Machinery, starting on page 317 written by B. Krishnamoorthi and R. C. Wood entitled "Time-Shared Operations with Both Interarrival and Service Times Exponential". It has been claimed by Heaps [10] that the effect of truncation actually outweighs most of the disadvantages. Most important, the use of truncation allows a save in storage and search time.

Perhaps it is worthwhile to note at this point that the effectiveness of a document retrieval system depends very much

on the selection of the data base. The suitability of CSDATA to fit the present system has been carefully studied. First of all, CSDATA is homogeneous in the sense that all its keywords are related to the field of computing science. Therefore, the occurrence of homonyms is very unlikely to happen. Since the system deals with semantic information, the exclusion of homonyms considerably simplifies the search process. Secondly, the terminologies of CSDATA are reasonably stable and not too specialized. Hence, the amount of keywords that may be ambiguously interpretated due to truncation is kept to a minimum. At the same time, no special treatment is required for any subset of keywords. Lastly, the collection is believed to be large enough to allow abstraction of significant data characteristics.

2.2 The Automatic Indexing Algorithm

As the system places greater emphasis on the output than the input, it is not necessary to develop a thesaurus. Instead, effort is devoted to generation of a comprehensive list of index terms. This is achieved by an automatic indexing algorithm based on the statistical association of terms within the document collection. It has been shown that terms which co-occur in document titles more frequently than the average are semantically, as well as statistically, related [11 to 17]. The set of index terms should satisfy the general goal of automatic indexing which is to provide a compact representation of the information content of the given data base.

To determine the importance of a term by means of statistical association techniques, it is necessary to make use of the co-occurrence frequency data, together with the total frequency data, in order to define some statistical association measures that reflect the degree of relatedness of the term with others. Before applying any association measure, it is desirable to exclude very high frequency terms (common terms) as well as very low frequency terms from the data base since such terms are semantically insignificant. Usually, a stop-list will serve this purpose.

Statistical association measures are generally expressed in terms of one or more of the following elements:

f_i = the frequency of occurrence of $term_i$,

f_{ij} = the frequency of co-occurrence of $term_i$ and $term_j$, and

n = the total number of documents in the data base.

Let w_{ij} = the indicator of the presence of $term_j$ in $document_i$.

$$= \begin{cases} 1 & \text{if } term_j \text{ is in } document_i, \\ 0 & \text{otherwise.} \end{cases}$$

Then, by definition,

$$f_j = \sum_{i=1}^{n} w_{ij}, \tag{2.1}$$

and, $$f_{ij} = \sum_{k=1}^{n} w_{ik} w_{kj}. \tag{2.2}$$

Suppose m is the total number of distinct terms in the data base, the matrix $W = (w_{ij})_{nxm}$ is often called the document-term matrix. Hence, f_i is equal to the j-th column sum of W, and f_{ij} is equal to the dot product of the transpose of the i-th column and the j-th column of W. Also, by definition, $f_{ij} = f_{ji}$ for all i, j = 1, 2, ... , m.

There are numerous statistical association measures that have been introduced into the literature of information storage and retrieval. A list of some of the most frequently used measures is given in Appendix B. Not surprisingly, each measure has been found to have its own merits and demerits. In fact, some measures are similar to one another in that they give equivalent rankings to the same set of terms [12]. Now, suppose that, according to some appropriate measure,

c_{ij} = the extent to which $term_i$ is associated with $term_j$ in the data base.

The matrix $C = (c_{ij})_{mxm}$ is often called a term-term association matrix. In the experiments to follow, three of the most common association measures are tested. They are :

(1) $$c_{ij} = f_{ij} / (f_i + f_j - f_{ij}). \qquad (2.3a)$$

(2) $$c_{ij} = f_{ij} / f_i^{1/2} f_j^{1/2}. \qquad (2.3b)$$

(3) $$c_{ij} = [f_{ij} - f_i f_j/n]^2/[f_i - f_i^2/n][f_j - f_j^2/n]. \qquad (2.3c)$$

It is convenient to define $f_{ii} = f_i$, so that $c_{ii} = 1$ for each of the above measures. It is obvious that each association measure gives rise to a symmetric term-term association matrix such that $c_{ij} = c_{ji}$ for all i, j = 1, 2, ... , m.

Now, the extent to which a term is associated with a document may be considered as the extent to which the term is associated with the terms in the document title. As a result, a term may bear a certain degree of relatedness to a document even though it does not appear in the document title. This peculiar feature can be interpreted as arising from the fact that different terms can have similar contexts and hence may be used as substitutes for others. In practice, these relations are used to aid indexing of new documents [17]. Let

g_{ij} = the extent to which term_j is associated with document_i.

= weighing factor for term_j in relation to document_i.

In accordance with the above view we may define

$$g_{ij} = \sum_{k=1}^{m} w_{ik}\, c_{kj} \Big/ \sqrt{\sum_{k=1}^{m} w_{ik}^2 \sum_{k=1}^{m} c_{kj}^2}\,. \qquad (2.4)$$

Note that $\sum_{k=1}^{m} w_{ik} = \sum_{k=1}^{m} w_{ik}^2$.

Now, let $W = (\underline{w}_1, \underline{w}_2, \ldots, \underline{w}_n)^T$ where $\underline{w}_i$, $i = 1, 2, \ldots, n$ represents the i-th row of W, and $(.)^T$ denotes the transpose of (.). Similarly, let $C = (\underline{c}_1, \underline{c}_2, \ldots, \underline{c}_m)$ where $\underline{c}_j$, $j = 1, 2, \ldots, m$ represents the j-th column of C. Then, equation (2.4) can be written in a more compact form as:

$$g_{ij} = \underline{W}_i \cdot \underline{C}_j \;/\; || \underline{W}_i || \; || \underline{C}_j ||, \tag{2.5}$$

where $||.||$ is the Euclidean norm. The matrix $G = (g_{ij})_{nxm}$ is called the weighted document-term matrix. In matrix notation, equation (2.5) becomes:

$$G = \mu WC, \tag{2.6}$$

where $\mu(i, j) = 1 \;/\; || \underline{W}_i || \; || \underline{C}_j ||$.

Suppose $G = (\underline{g}_1, \underline{g}_2, \ldots, \underline{g}_m)$ where $\underline{g}_j$, $j = 1, 2, \ldots, m$ represents the j-th column of G. Then, according to the above assumption, the elements of G represent the extent to which a term is related to a document in the data base. The measure of the extent to which a term_j is related to all documents can be defined as:

$$y_j = || \underline{g}_j ||.$$

$$= [\sum_{i=1}^{n} g_{ij}^2]^{1/2}. \tag{2.7}$$

= the significance value of term_j in the data base.

In order to determine the set of index terms that carries significant information content, an arbitrary cut-off value K is imposed. The average of all y_j, $j = 1, 2, \ldots, m$ seems to be a reasonable cut-off value. Hence, by definition,

$$K = (1/n) \sum_{j=1}^{n} y_j . \tag{2.8}$$

Consequently, every term t_j such that $y_j \geq K$ will be regarded as an index term. Suppose there are m' number of such terms which constitute the set of index terms I, then, in set notation,

$$\begin{aligned} I &= \{\text{Set of index term}\}. \\ &= \{t_i : y_i \geq K \text{ for all } i = 1, 2, \ldots, m'\}. \end{aligned} \tag{2.9}$$

The automatic indexing algorithm is summarized in the following statements:

1. Create document-term matrix W.
2. Generate term-term association matrix C using an appropriate statistical association measure.
3. Calculate $G = {}_{\mu}WC$ to form the weighted document-term matrix.
4. Calculate $y_j = ||\underline{g}_j||$ for all $j = 1, 2, \ldots, m$.
5. Calculate K and determine the set of index terms I.

2.3 Storage Problems

A subset of 5150 journal articles is taken from CSDATA for testing the automatic indexing algorithm. It can be seen from the flowchart of Fig. 2.2 that the procedures are rather straightforward. Nevertheless, a complication arises as very large matrices are involved in computations at various stages of the algorithm. There are altogether 1801 distinct terms in the test data. Thus, a document-term matrix alone will require approximately 37 million bytes of storage. Obviously, the conventional method of storage and matrix multiplication cannot be used. It is therefore necessary to develop an appropriate technique to cope with the problems created by such matrices.

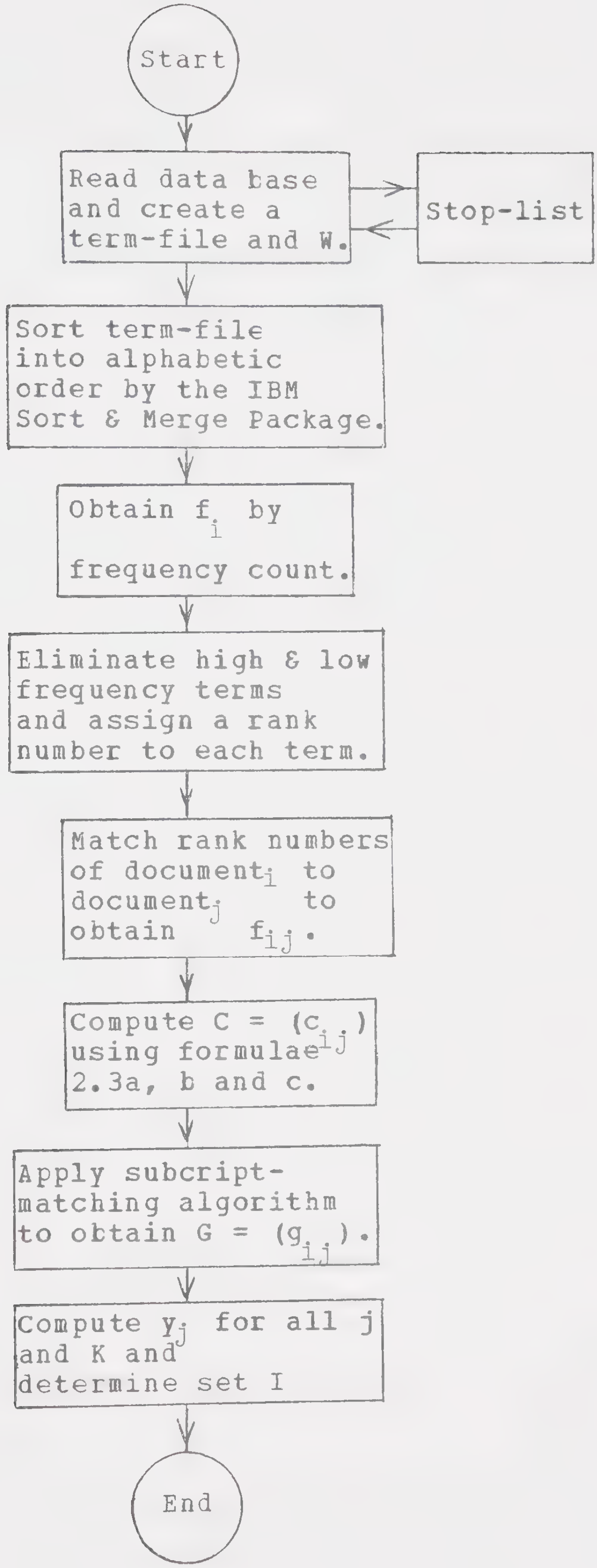


Fig.2.2 Flowchart for the Automatic Indexing Algorithm

The simplest approach is to store the elements of these matrices in the most economical storage format; then matrix multiplication can be carried out by applying a fairly simple subscript-matching algorithm [18]. Consider the matrices W, C, and G. Since they are very sparse matrices, only non-zero elements need be stored. In order that the original matrix can be restored efficiently, the row number, the number of non-zero elements in the row, the column number and the corresponding value for each non-zero element are stored. The storage format is shown in Fig. 2.3. It is known as the least-storage scheme.

i	p_i	j_1	v_{ij_1}	j_2	v_{ij_2}	. . .	j_{p_i}	$v_{ij_{p_i}}$

Fig. 2.3 The Least-storage Scheme

The example in Fig. 2.3 records p_i number of non-zero elements in row i. For a matrix of dimension nxm, we have the following interpretation of symbols:

i = i-th row indicator, $1 \leq i \leq n$,

p_i = number of non-zero elements in row i, $0 \leq p \leq m$,

j_α = α-th column indicator, which points to the column in the original matrix, $\alpha = 1, 2, \ldots, p_i$; $1 \leq i \leq n$; $1 \leq j_\alpha \leq m$,

v_{ij_α} = value of the element of the i-th row and the j_α-th column

of the original matrix.

The number of non-zero elements in a row, p_i is included to facilitate the retrieval of all the row elements. This can be achieved by the following FORTRAN statement:

```
READ(ID,NF) I,PI,((JPI(J),V(I,J)),J=1,PI)
```

where ID is the input device number, NF is the format number and PI is an integer variable. An example is given below to illustrate how the least-storage scheme actually works. Suppose a matrix A is given by:

$$A = \begin{bmatrix} 4 & 5 & 0 & 0 & 4 & 0 & 0 \\ 0 & 7 & 6 & 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 8 & 0 & 0 & 0 & 0 & 2 & 3 \\ 0 & 0 & 7 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Then, the entire matrix will be stored in a sequential file as:

1 3 1 4 2 5 5 4	2 3 2 7 3 6 6 9	3 1 4 1	4 3 1 8 6 2 7 3	5 1 3 7

Thus, to represent row 2, for example,

i = i-th row = 2,

p_i = number of non-zero elements in row 2 = 3,

j_1 = first non-zero element in row 2 occurs under column 2,

$v_{ij_1} = a_{22} = 7$,

j_2 = second non-zero element in row 2 occurs under column 3,

$v_{ij_2} = a_{23} = 6$,

j_3 = third (last) non-zero element in row 2 occurs under column 6,

$v_{ij_3} = a_{26} = 9$.

It is fairly easy to show that the least-storage scheme requires much less storage than the conventional method which stores every single element of the matrix. Given an nxm sparse matrix, suppose

S_c = the total number of words required to store the given matrix by the conventional method, and

S_ℓ = the total number of words required to store the given matrix by the least-storage scheme.

Assuming that at least one non-zero element appears in each row or column of the matrix, then, by definition,

$$S_c = nm, \tag{2.10}$$

and,
$$S_\ell = 2n + 2 \sum_{i=1}^{n} p_i ,$$
$$= 2n(1+h), \tag{2.11}$$

where h = the average number of non-zero elements in each row of the matrix. It can be shown that for a given n, $S_c > S_\ell$ for every $m > 2(1+h)$. Graphically, this is shown in Fig. 2.4.

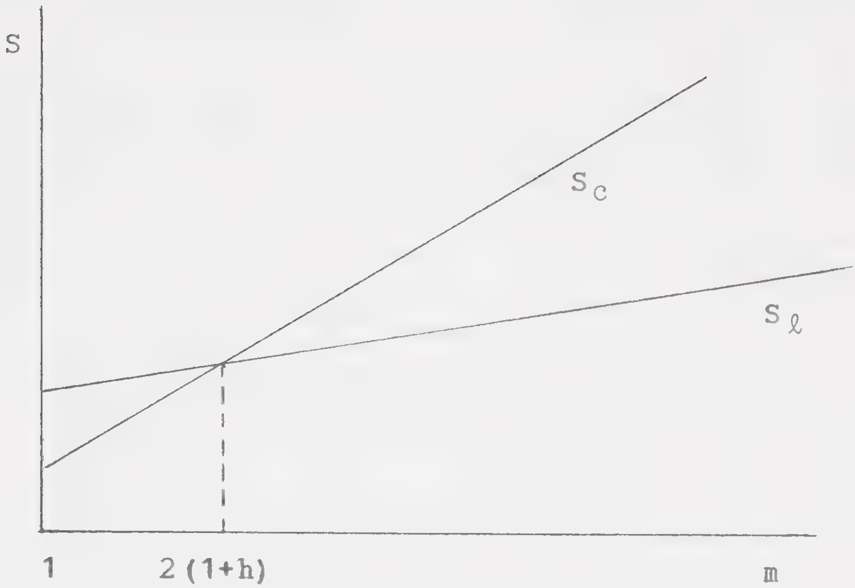


Fig. 2.4 Comparing Storage Requirements for the Conventional Storage Scheme and the Least-storage Scheme for Ordinary Matrices

In CSDATA, the average number of terms per document is approximately six. Hence, to store the document-term matrix by using the least-storage scheme requires about 1/150 of the storage required by the conventional method. It may also be noted that for a sparse symmetric matrix such as the term-term association matrix, the storage requirement can further be reduced by storing only the diagonal and upper (or lower) triangular non-zero elements. When using the least-storage scheme, care must be taken to note that for certain rows (columns) the diagonal and the upper (or lower) triangular elements may all be equal to zero, then the record for this row (column) will not appear in the storage file. In this case, such

a row (column) is said to be null with respect to the storage file.

As an illustration, consider the sparse symmetric matrix A given by:

$$A = \begin{bmatrix} 3 & 0 & 0 & 0 & 5 & 0 & 0 \\ 0 & 7 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 6 & 8 & 0 & 0 & 0 \\ 0 & 0 & 8 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 6 & 0 & 9 \\ 0 & 4 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 9 & 0 & 5 \end{bmatrix}$$

Then, the entire matrix will be stored in a sequential file as:

1 2 1 3 5 5	2 2 2 7 6 4	3 2 3 6 4 8	5 2 5 6 7 9	6 1 6 1	7 1 7 5

Note that the fourth row (column) of A is null w.r.t. The storage file.

Consider a general mxm sparse symmetric matrix. Let

S_c' = the total number of words required to store the diagonal and upper triangular elements of the given matrix by the conventional method.

S_ℓ' = the total number of words required to store the diagonal and upper triangular elements of the given matrix by the least-storage scheme.

Then, by definition,

$$S_c' = m(m+1)/2, \tag{2.12}$$

and,

$$S_\ell' = 2m' + 2 \sum_{i=1}^{m'} p_i ,$$

$$= 2m'(1+h'), \tag{2.13}$$

where m' = number of rows with $p_i > 0$, $i = 1, 2, \ldots , m$; $m' \leq m$,

h' = the average number of non-zero elements in each row of the matrix for which $p_i > 0$, $i = 1, 2, \ldots , m$.

Therefore,

$$S_\ell' \leq 2m(1+h'). \tag{2.14}$$

It can be seen that $S_c' > S_\ell'$ for every $m > 3+4h'$. Graphically, this is shown in Fig. 2.5.

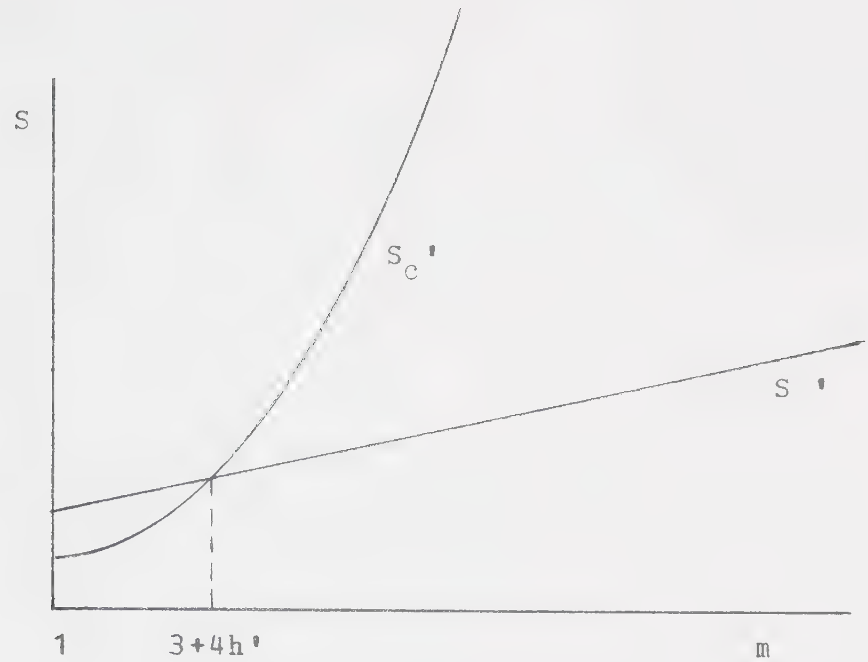


Fig. 2.5 Comparing Storage Requirements for the Conventional Storage Scheme and the Least-storage Scheme for Symmetric Matrices

Thus the least-storage scheme allows a saving of a tremendous amount of storage space. In many instances, the matrix stored in this format can be loaded in core, thereby eliminating costly and time-consuming I/O access times that would be required if the matrix were stored on an auxiliary storage device. A very simple subscript-matching algorithm has been devised in conjunction with the least-storage scheme in order that matrix multiplications can be carried out effectively and efficiently.

2.4 Programming Considerations

In this section, some programming details for generating the set of index terms will be discussed briefly. Several intermediate files are essential for the entire process.

(i) Data File:

The data base is read sequentially. Each document is assigned a document number and each title term a position number according to its sequence of occurrence. For each term of each document, a record is written in the format as shown in Fig. 2.6.

Document No.	Position No.	Term

Fig. 2.6 Record Format of Data File

The set of sequential records constitutes the data file. Note that this file preserves the original information of the data base.

(ii) Document-term File:

The document-term file is the data file arranged in alphabetical order according to terms. This file is also called the inverted-index file.

(iii) <u>Frequency File:</u>

The frequency file consists of the frequency f_i for all distinct terms in the data base. This is easily created by counting the number of records that contain the term. A record of the frequency file is shown in Fig. 2.7.

Term_i	Frequency_i

Fig. 2.7 Record Format of Frequency File

(iv) Term-directory File:

The frequency file is sorted in descending order of frequency. High and low frequency terms are eliminated. Each term is then assigned a rank number according to its sequence in the sorted frequency file. In the case when several terms have the same frequency of occurrence, consecutive numbers are assigned arbitrarily. A record of the term-directory file is shown in Fig. 2.8.

Rank No._i	Term_i	Frequency_i

Fig. 2.8 Record Format of Term-directory File

(v) Term-pair File:

From the data file, all possible term pairs are abstracted from each record to create the term-pair file. A record of the term-pair file is shown in Fig. 2.9.

Rank No. of Term_i	Rank No. of Term_j	Term_i	Term_j

Fig. 2.9 Record Format of Term-pair File

(vi) Co-occurrence Frequency File:

The term-pair file is modified by interchanging the terms in any term pair whose second term has lower alphanumeric value than its first term. The file is then sorted alphabetically according to the term pairs. The frequency of co-occurrence of each term pair is then counted. A record of the co-occurrence frequency file is shown in Fig. 2.10.

Rank No. of Term_i	Rank No. of Term_j	Term_i	Term_j	f_{ij}

Fig. 2.10 Record Format of Co-occurrence Frequency File

(vii) Term-term Association File:

Three different term-term association measures are used. Note that since C is symmetric, only diagonal and upper diagonal non-zero elements are stored. A record of the term-term association file is shown in Fig. 2.11 where $c_{ij}^{(k)}$, k = 1, 2, 3 refers to the equations of (2.3a), (2.3b), and (2.3c).

Rank No. of $Term_i$	Rank No. of $Term_j$	$Term_i$	$Term_j$	$c_{ij}^{(1)}$	$c_{ij}^{(2)}$	$c_{ij}^{(3)}$

Fig. 2.11 Record Format of Term-term Association File

(viii) Weighted Document-term File:

The document-term file and the term-term association file are used to form the weighted document-term file. Rank numbers are used to represent subscripts of the elements of the various matrices. A record of the weighted document-term file is shown in Fig. 2.12 where $g_{ij}^{(k)}$, k = 1, 2, 3 corresponds to the respective measures of $c_{ij}^{(k)}$, k = 1, 2, 3.

Rank No. of $Term_i$	Rank No. of $Term_j$	$Term_i$	$Term_j$	$g_{ij}^{(1)}$	$g_{ij}^{(2)}$	$g_{ij}^{(3)}$

Fig. 2.12 Record Format of Weighted Document-term File

(ix) Index-term File:

For each weighted document-term file, the respective cut-off value according to (2.8) is calculated and an index-term file is determined.

The generation of all these files are quite simple except for the weighted document-term file. The document-term file and the term-term association file are transformed into the least-storage format so that the subscript-matching algorithm can be applied. The algorithm is discussed in general below.

Consider an mxn large general sparse matrix $B = (b_{ij})$ and an nxn large symmetric sparse matrix $C = (c_{ij})$. Let B and C be stored according to the least-storage algorithm as two separate files, respectively called File B and File C. It is required to calculate the product of B and C. Let $A = (a_{ij})$, $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. Then $A = B \times C$ is given by

$$a_{ij} = \sum_{k=1}^{n} b_{ik}c_{kj}, \tag{2.15a}$$

$$= \sum_{k=1}^{n} b_{ik}c_{jk}. \tag{2.15b}$$

Suppose the elements of the i-th row of B constitute a record in File B as denoted by the symbols $(i, p_i, \{(r_\alpha, b_{ir_\alpha}), \alpha = 1, 2, \ldots, p_i\})$; and the elements of the j-th column of C constitute a record in File C as denoted by the symbols $(j, q_j, \{(s_\beta, c_{js_\beta}), \beta = 1, 2, \ldots, q_j\})$. Then, a_{ij}

is the sum of the product of all b_{ir_α} and c_{js_β} in which $r_\alpha = s_\beta$. There are three possible cases.

Case I:

If the j-th column of C is not null with respect to File C and $r_1 \geq s_1$, then for each match such that $r_t = s_v$ where $t \in \{\alpha\}$, $v \in \{\beta\}$, calculate the product $b_{ir_\alpha} c_{js_\beta}$. The sum of all these products yields a_{ij}. Diagrammatically, the matching mechanism is as shown in Fig. 2.13.

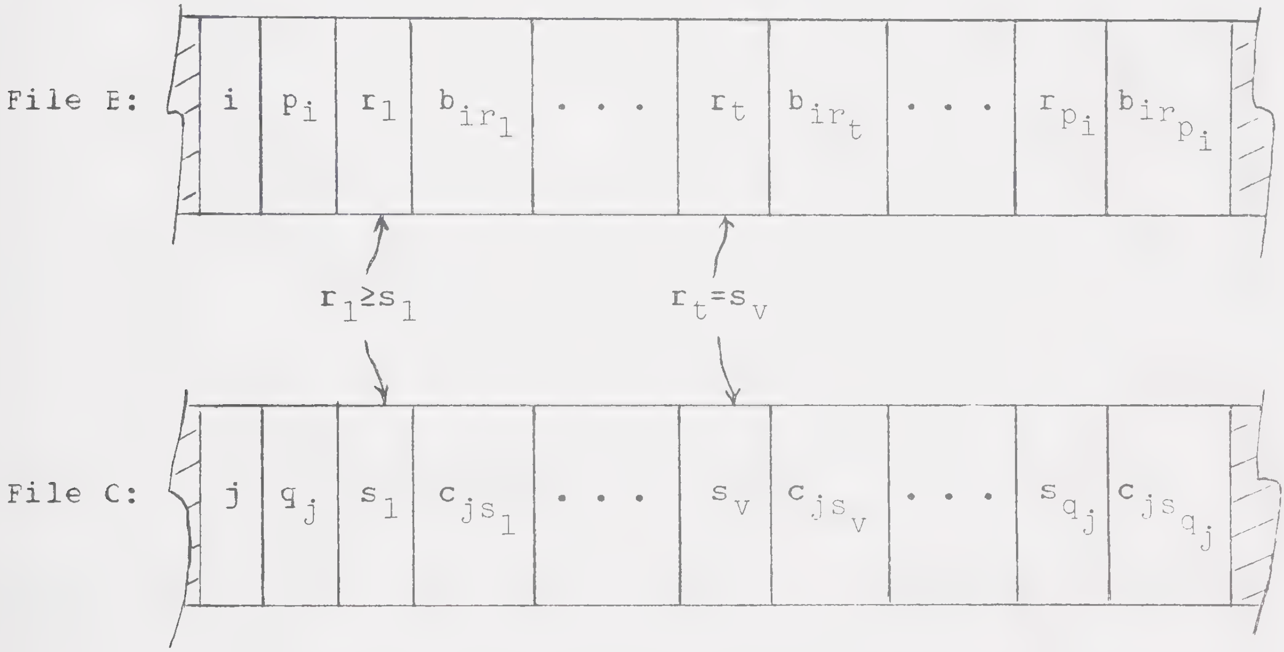


Fig. 2.13 Matching Mechanism (Case I)

Case II:

If the j-th column of C is not null with respect to File C and $r_1 < s_1$, then for each r_t, $t = 1, 2, \ldots, p_f$ where $p_f \leq p_i$ and $r_t < s_1$, match the subscripts of the elements of the r_t-th column of C (ignore if null with respect to File C). For each

$s_v = j$, $v = 1, 2, \ldots, q_{r_t}$, calculate the product $b_{ir_t} c_{r_t s_v}$. Now, for all r_δ, $\delta = p_f+1, p_f+2, \ldots, p_i$; $r_\delta \geq s_\gamma$, $\gamma = 1, 2, \ldots, q_j$. Hence, the matching and calculating processes are the same as Case I. The sum of all these products yields a_{ij}. Diagrammatically, the matching mechanism is as shown in Fig. 2.14.

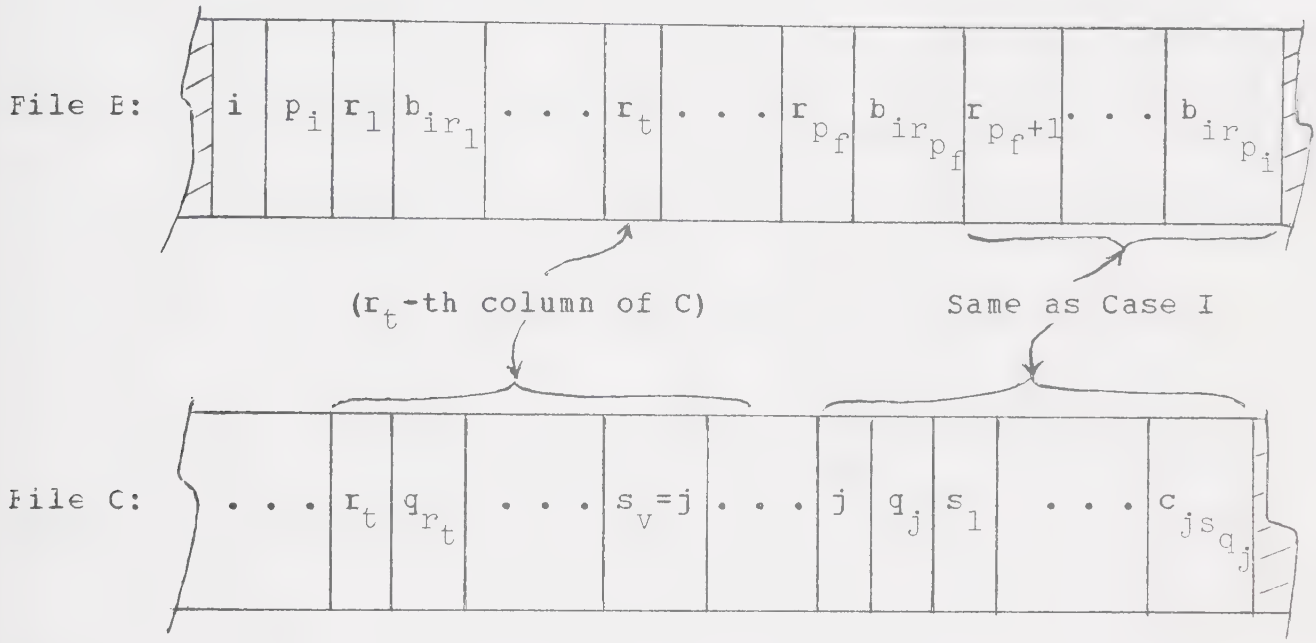


Fig. 2.14 Matching Mechanism (Case II)

Case III:

If the j-th column of C is null with respect to File C, then for each r_t, $t = 1, 2, \ldots, p_g$, $p_g \leq j-1$, match the subscripts of the elements of each r_t-th column of C (ignore if null with respect to File C). For each $s_v = j$, $v = 1, 2, \ldots, q_{r_t}$, calculate the product $b_{ir_t} c_{r_t s_v}$. The sum of all these products yeilds a_{ij}. Diagrammatically, the matching mechanism is as shown in Fig. 2.15.

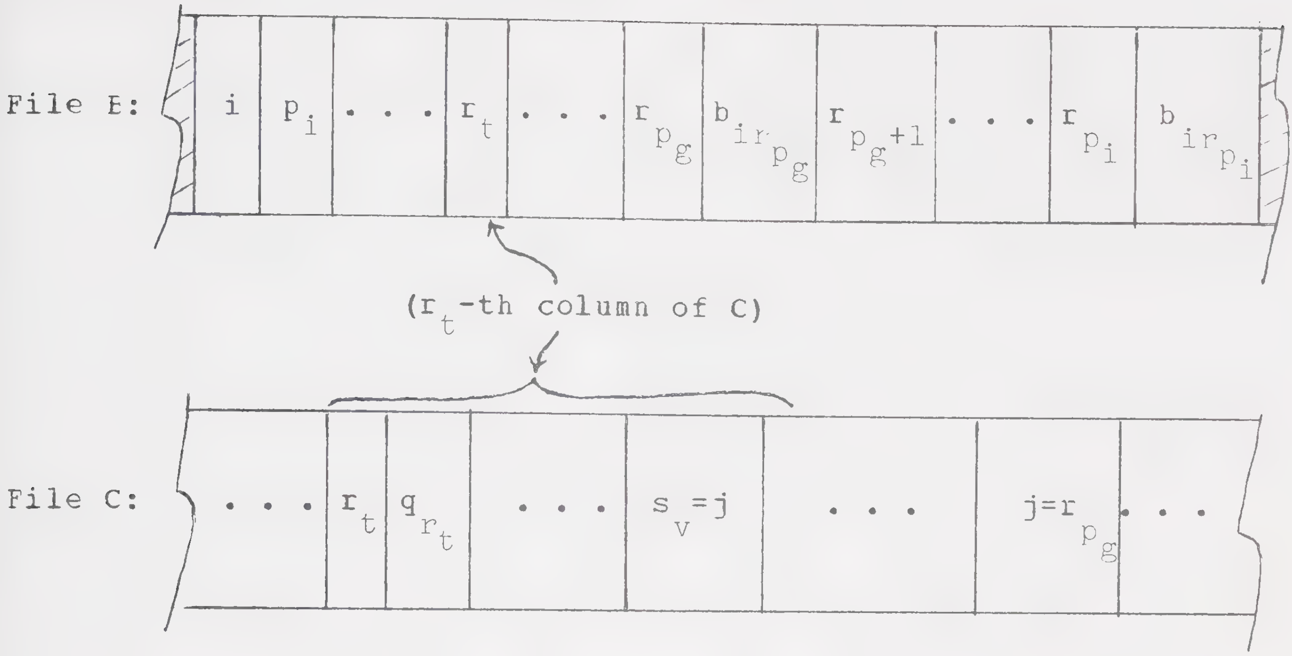


Fig. 2.15 <u>Matching Mechanism (Case III)</u>

Finally, the subscript-matching algorithm is presented formally in the following manner:

<u>STEP</u> 1: For each i, $i = 1, 2, \ldots, m$, obtain from File B $(i, p_i, \{(r_\alpha, b_{ir_\alpha}), \alpha = 1, 2, \ldots, p_i\})$. At end, go to Step 7.

<u>STEP</u> 2: For each j, $j = 1, 2, \ldots, n$, obtain from File C $(j, q_j, \{(s_\beta, c_{js_\beta}), \beta = 1, 2, \ldots, q_j\})$. At end, go to Step 1.

<u>STEP</u> 3: If the j-th column of C is null with respect to File C, go to Step 6. If $r_1 < s_1$, go to Step 5. Otherwise, go to Step 4.

<u>STEP</u> 4: (Case I) For each $r_t = s_v$, $t = 1, 2, \ldots, p_i$; $v = 1, 2, \ldots, q_j$, calculate the product $b_{ir_t} c_{r_t s_v}$. Then

$a_{ij} = \sum_{r_t = s_v} b_{ir_t} c_{js_v}$. Go to Step 2.

STEP 5: (Case II) For each $r_t < s_1$, $t = 1, 2, \ldots, p_i$, $p_f \leq p_i$, get the r_t-th column of C (ignore if null with respect to File C). For each $s_v = j$, $v = 1, 2, \ldots, q_{r_t}$, calculate the product $b_{ir_t} c_{r_t s_v}$. Then, for each $r_\delta \geq s_\gamma$, $\delta = p_f+1, p_f+2, \ldots, p_i$; $\gamma = 1, 2, \ldots, q_j$, calculate the product $b_{ir_\delta} c_{js_\gamma}$ for each $r_\delta = s_\gamma$. Then, $a_{ij} = \sum_{\substack{s_v = j \\ r_t < s_1}} b_{ir_t} c_{r_t s_v} + \sum_{r_\delta = s_\gamma} b_{ir_\delta} c_{js_\gamma}$. Go to Step 2.

STEP 6: (Case III) For each r_t, $t = 1, 2, \ldots, p_g$, $p_g \leq j-1$, get each r_t-th column of C (ignore if null with respect to File C). For each $s_v = j$, $v = 1, 2, \ldots, q_{r_t}$, calculate the product $b_{ir_t} c_{r_t s_v}$. Then, $a_{ij} = \sum_{s_v = j} b_{ir_t} c_{r_t s_v}$. Go to Step 2.

STEP 7: STOP.

The flowchart as shown in Fig. 2.16 describes the subscript-matching algorithm.

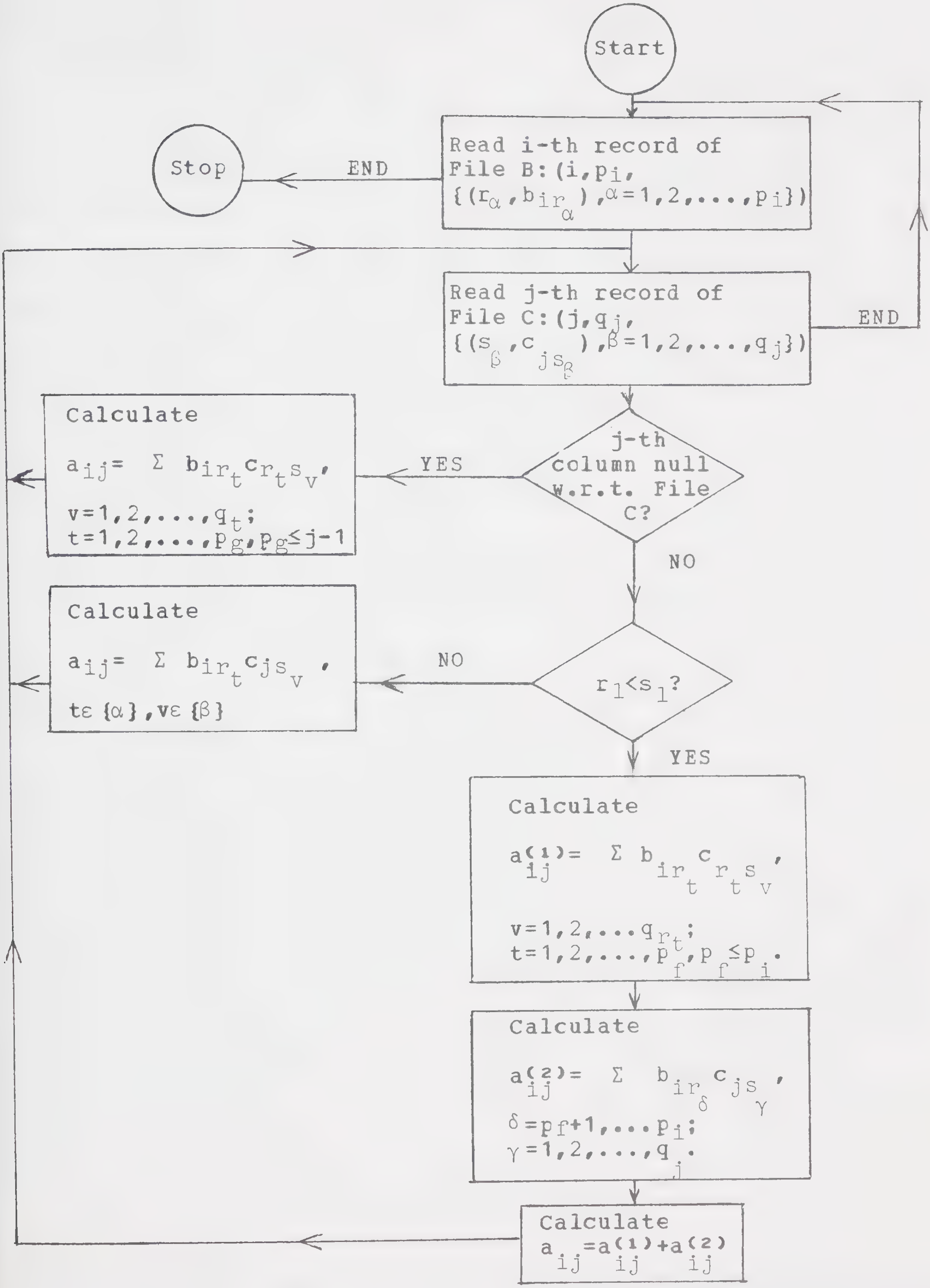


Fig.2.16 Flowchart for the Subscript-matching Algorithm

2.5 The Index Term List

There are a total of 29,677 terms in the test data of 5,150 documents but there are a total of only 3,787 distinct terms. The distribution of terms in documents is shown in the graph of Fig. 2.17. It is found that the average number of terms per document is approximately 5.76. The distribution of the number of documents that contain a given term is shown in the graph of Fig. 2.18. The average number of documents per term is approximately 32.65. After eliminating non-significant terms by a stop-list and excluding terms of low frequency (frequency = 1), a total of 1,801 distinct terms are left to be processed. Similarly, there are 84,450 possible term pairs but only 11,038 pairs are used.

It is realized that since the matrices are very large, the calculation of $G = \mu WC$ will require a tremendous amount of computing time. Therefore, several random samples of different sample sizes are tested by calling the IBM Pseudo Random Number Generator subroutine CS003A which is written in FORTRAN IV. Each document has been assigned a document number, and uniformly distributed pseudo random numbers in the closed interval [0,5150] are generated by the following calling sequence:

```
  CALL CS003A(INIT)
  DO 1 I=1,J
1 CALL CS003C(A,B,SIZE,N)
```

where INIT is a positive odd integer input value to initialize the algorithm,

J is the sample size,

[A, B] = [0, 5150] are real input parameters for the lower and upper limits of interval,

SIZE is the random number returned by CS003C,

N is the sequence number.

Three samples of size 50, 100 and 200 were tested. As the sample size increased, the sets of index terms resulting from the samples were found to converge to the same limiting set of index terms. The index term lists that result by using measures (2.3a), (2.3b) and (2.3c) respectively contain 815, 816 and 823 different index terms; all the index terms that appear in the first list also appear in the other two, and they share approximately the same rank in each case. It can thus be concluded that the set of index terms common to all lists is representative of the significant terms of the data base. The final set of index terms is then chosen to be the intersection of the sets resulted from the three different index term lists using a sample size of 200. The corresponding significance values are taken to be the mean of the corresponding three significance values. The set of significance values are further normalized into the interval [0, 1], and will eventually constitute a subset of the feedback control parameters. The three different sets of index terms and the final set of index terms together with the significance values are given in Appendix C and Appendix D, respectively.

Fig. 2.17 Graph of Distribution of Terms in Documents

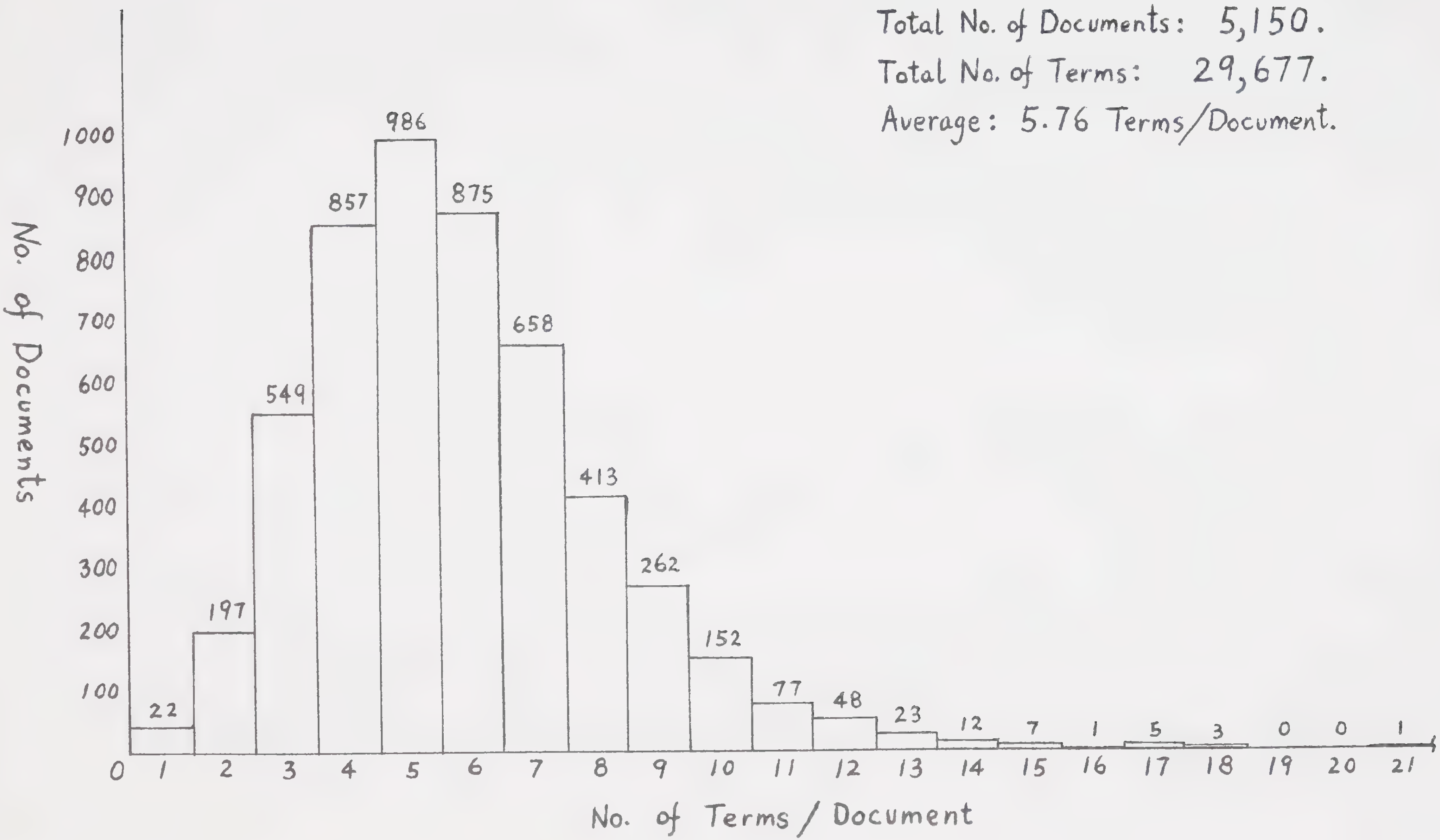
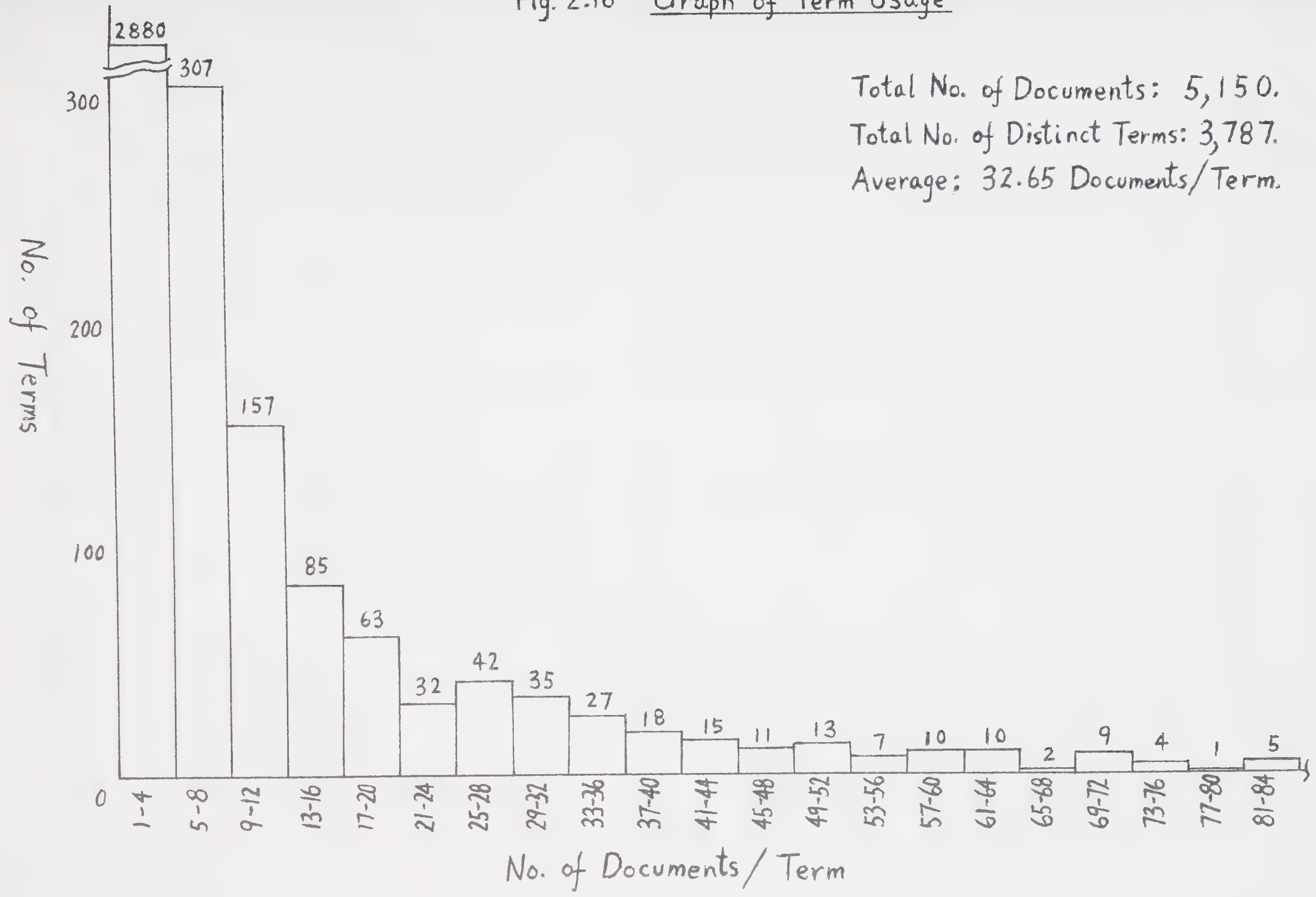


Fig. 2.18 Graph of Term Usage



Total No. of Documents: 5,150.

Total No. of Distinct Terms: 3,787.

Average: 32.65 Documents/Term.

CHAPTER III

DEFINITIONS OF SYSTEM FUNCTIONS

3.1 The Query Language

For any document retrieval system, it is essential to formulate a query language designed to convey exactly the user's information need to the search processor. The transformation process is based on some logic operations implicitly contained in the query language. In fact, what constitutes the core of the search logic and the query language is the set of logic operators used to formulate the search requests. Hence, it is extremely important that the syntax of the query language be well-defined. Conventionally, the Backus Normal Form (BNF) is used. Table 3.1 gives a brief explanation of the BNF symbols.

Symbol	Meaning
< >	variable name or expression
:: =	is defined to be
$\{\ \}_a^b$	repeat m number of times, where $m \in [a, b]$, a, b being integers.
|	exclusive OR
ƀ	blank

Table 3.1 Interpretation of BNF Symbols

The following specifications in BNF represent the syntax of the query language connected to the present system:

<search request> :: = <question statement>{<parameter>}$^{8}_{1}$
<end statement>

<question statement> :: = QUEb̸<remark><relevance estimate> |
QUEb̸<relevance estimate> | <comment statement>
<question statement>

<parameter> :: = <leading statement>{<subsequent statement>}$^{9}_{0}$

<leading statement> :: = <comment statement><leading statement> |
<primary logic operator>b̸<search particulars><weight>

<primary logic operator> :: = AND | NOT

<search particulars> :: = <search type>b̸<search item>

<search type> :: = <author> | <coden> | <title term> | <year>

<author> :: = A

<coden> :: = C

<title term> :: = T

<year> :: = Y

<search item> :: = item to be searched.

<weight> :: = {<decimal digit>}$^{4}_{0}$

<decimal digit> :: = 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<comment statement> :: = b̸b̸b̸<remark>

<remark> :: = a string of symbols.

<subsequent statement> :: = <secondary logic operator>b̸<search
particulars><weight> | <comment statement>
<subsequent statement>

<secondary logic operator> :: = b̸OR | NOR

<relevance estimate> :: = <recall estimate><precision estimate>

<recall estimate> :: = <weight> | <recall estimate>

<precison estimate> :: = <weight> | <precision estimate>

<end statement> :: = END | ENDb̸<remark>

The search processor may be so implemented so that it is capable of performing a search for either a single search request or a batch of search requests. This is enabled by the definition:

<batch> :: = {<search request>}$_1^n$,

where n is the maximum number of search requests the system can handle. Since a sequential search technique is used, the advantage of batching is obviously a considerable reduction of search time. In order that no error in a search request may affect other members in the batch, those incorrect search requests are treated as if they do not belong to the batch. Upon output, appropriate error messages are issued.

In a batch of search requests, the QUEstion statement and the END statement of each request serve as delimiters. Each request allows up to eight parameters. Each parameter is led by a statement using one of the primary logic operators AND or NOT. It is then followed by not more than nine other statements in any combination of the secondary logic operators b̸OR and NOR.

Four search types can be used. They are author name, journal coden name, title term and year of publication, respectively denoted by A, C, T, and Y. Each search request item may be given an arbitrary term weight, all up to four digits. In the absence of assignment of weight, the default value of one is automatically assigned. At the same time, the user may specify on each QUEstion statement his anticipated recall and precision values as defined in Section 5.1 which have a default of one

hundred percent. The term weights and the estimated recall and precision values will eventually constitute a subset of the feedback control parameters. It is noted that any number of comment statements may appear anywhere in a search request. They do not contribute to the search operations, but are merely designed for the users to make remarks.

A user may submit his batch of search requests either in the form of a deck of cards or via a terminal. In any case, the appropriate input format must be used for the different kinds of statements. The input formats can be generally classified into four types as shown in Fig. 3.1 (a) to (d).

1 3	4	5	65 68	69 72	73 80
QUE	blank	Comment	Recall Estimate	Precision Estimate	

(a) <u>Type I: QUEstion Statement</u>

1 3	4 80
ƀƀƀ	Comment

(b) <u>Type II: Comment Statement</u>

1 3	4	5	6	7 68	69 72	73 80
Logic	blank	Search Type	blank	Search Item	Term Weight	

(c) <u>Type III: Logic Statement</u>

1 3	4	5 80
END	blank	Comment

(d) <u>Type IV: END Statement</u>

Fig. 3.1 <u>Search Request Input Format</u>

Note that columns 73-80 of any input format type can be used freely by the user. Sometimes, sequence numbers or identifiers may prove to be useful. The content of these columns are ignored by the search processor.

The following is an example of a batch of two search requests.

```
QUE SAMPLE REQUEST #1                                  90  85
   SUBMITTED BY USER ALO.
AND A ABDALI SK                                           120
OR A LEVIALDI S
   TOPIC OF INTEREST IS PICTURE PROCESSING.
AND T PICTURE                                              50
AND T PROCESSING
   NOT LIKELY TO APPEAR IN THE YEARS 68 TO 70.
NOT Y 68
NOT Y 69
 OR Y 70
   CODEN NAME OF JOURNAL IS CACMA OR PAT.
AND C CACMA                                                30
 OR C PAT                                                  40
END

   SAMPLE REQUEST #2.
QUE
   REQUIRE ALL ARTICLES BY G. SALTON,
   APPEAR IN THE JOURNAL IFSRA IN 1971.
AND A SALTON G
AND C IFSRA
AND Y 71
   ALSO REQUIRE A PAPER BY A. ROSENFELD,
   APPEAR IN THE JOURNAL JACOA IN 1966.
AND A ROSENFELD A
AND Y 66
   END OF SAMPLE REQUEST #2.
END OF BATCH OF TWO SEARCH REQUESTS.
```

The first example requests a weighted search for any document written by ABDALI SK or LEVIADI S on the subject matter of PICTURE PROCESSING and appearing in the journal called the Communications of the Association for Computing Machinery (CACMA) or the journal called Pattern Recognition (PAT), not from 1968 to 1970. The second example specifically requests all the articles written by SALTON G and appearing in the journal called Information Storage and Retrieval, in 1971. It also requests a paper written by ROSENFELD A appearing in the Journal of the Association for Computing Machinery, 1966.

3.2 Relevance Criteria

As stated in the formal query language definition, the user may assign weights to the terms that comprise his queries. These weights are designed to reflect the user's own point of view about the term usage or subject matter. A weak point in this approach is that the user often does not have the slightest idea how much weight he needs to assign to a term and how relative the weights should be in order that the system will interpret his viewpoints correctly. In some instances, a term considered to be important to the user may be very insignificant to the system. To remedy this, the list of index terms and their significance values are presented to the user to assist his preliminary judgment of the importance of terms. However, relevance judgment of the set of retrieved documents depends also on some system parameters. The set of relevance judgment criteria that takes into account both the user's viewpoint and

the system parameters may be developed in general as follows.

Let $D = \{\underline{d}_1, \underline{d}_2, \ldots, \underline{d}_n\}$ be a set of n documents that constitutes the data base. Suppose m is the number of distinct terms used to index all the documents. Then, these index terms can be regarded as defining an m-dimensional term space denoted by T^m. Each document $\underline{d}_i$, i = 1, 2, , ... , n is then represented by a vector or an m-tuple $(d_1(i), d_2(i), \ldots, d_m(i))$ in T^m in which each $d_j(i)$, j = 1, 2, ... , m is the significance value of the j-th term in the i-th document. Furthermore, the significance value of a term is bounded on [a, b] where $b > a > 0$ are real numbers.

Suppose a query is denoted by $\underline{q} = (q_1, q_2, \ldots, q_m)$ where each q_j, j = 1, 2, ... , m is the weight attached to the j-th term corresponding to T^m and is bounded on [a, b]. These weights may be assigned manually or may be a result of automatic adjustment. Again, $\underline{q}$ may be regarded as a vector or m-tuple in T^m. It is then possible to define some criteria to determine whether any given document $\underline{d}_i$ is relevant to $\underline{q}$. This set of criteria is known as the relevance judgment criteria or simply the relevance criteria. These criteria are given in Definition 3.1.

Definition 3.1

Suppose ω, σ, ξ, δ are arbitrarily small, positive, real numbers, then a document $\underline{d}_i$ is said to be relevant to a given query $\underline{q}$ if and only if one of the following conditions is

satisfied:

1. $\underline{d}_i = \underline{q}$. (3.1)

2. $| d_j^{(i)} - q_j | \leq \omega$, for all $j = 1, 2, \ldots, m$. (3.2)

3. $|| \underline{d}_i - \underline{q} || \leq \sigma$. (3.3)

4. $\beta_1(1-\underline{d}_i \cdot \underline{q}/|| \underline{d}_i || \; || \underline{q} ||) + \beta_2 \Big| || \underline{d}_i || - || \underline{q} || \Big| \leq \xi$,

   where $\beta_1, \beta_2 \in [0, 1]$ are real constants. (3.4)

5. $1 - \tau(n_1/n_2) \leq \delta$,

   where n_1 = number of common terms used to index $\underline{d}_i$ and $\underline{q}$,

   n_2 = number of different terms used to index $\underline{d}_i$ and $\underline{q}$,

   τ = a real constant. (3.5)

Obviously, condition one is the most desirable relevance criterion of all since the document $\underline{d}_i$ is exactly specified by the query $\underline{q}$ and perfect matching occurs in that $d_j^{(i)} = q_j$ for all $j = 1, 2, \ldots, m$. However, in general, this condition is too restrictive for practical considerations. Therefore, some tolerance values are introduced to allow more flexible judgment of relevance. Condition two implies that if each absolute value of the difference of the significance value of $\underline{d}_i$ and the corresponding weight of $\underline{q}$ is less than or equal to a given tolerance value ω, then the document $\underline{d}_i$ is taken to be relevant to the query $\underline{q}$. In the case when all these absolute values are

equal to zero, condition one is maintained. Similarly, condition three promises that if the length of the vector difference of $\underline{d}_i$ and $\underline{q}$ is less than or equal to a given tolerance value σ, then the document $\underline{d}_i$ will be regarded as relevant to the query $\underline{q}$. If the value of $\| \underline{d}_i - \underline{q} \|$ is zero, condition one is again maintained.

The inclusion of β_1 and β_2 in condition four allows varying stress of importance upon either the angle between $\underline{d}_i$ and $\underline{q}$ or the absolute value of the difference in lengths of the two vectors. The usefulness of β_1 and β_2 will be seen in section 3.3. Lastly, condition five merely states that if the number of common terms used to index $\underline{d}_i$ and $\underline{q}$ is to some degree close to the number of different terms used to index $\underline{d}_i$ and $\underline{q}$, then the document $\underline{d}_i$ can be regarded as relevant to the query $\underline{q}$. This criterion, as well as criteria two, three and four, may result in judging the document $\underline{d}_i$ relevant even though some of the terms used in $\underline{d}_i$ do not appear in the query $\underline{q}$ but are actually related in context to it.

3.3 A Relevance Measure

According to the set of relevance criteria of Definition 3.1, it is possible to have several documents judged as relevant to a given query. Hence, it is desirable to have some kind of

relevance measure which can distinguish the more relevant documents from the less relevant ones. Recall that (3.4) provides an option for emphasis on either the angle between $\underline{d}_i$ and $\underline{q}$ or the absolute value of the difference in lengths of the two vectors. In one case, β_1 and β_2 can both be set equal to one-half to indicate that the two quantities are equally important. It is, of course, possible to have any combination of values for β_1 and β_2. The determination of β_1 and β_2 depends, for example, on the definition of $\underline{d}_i$ and $\underline{q}$, the relationship between the two sets of weights, and so forth. For instance, under the above definition of $\underline{d}_i$ and $\underline{q}$, one may set (β_1, β_2) to be (0, 1) since if the coordinates of the two vectors are close together, then the angle between the two vectors must tend to be zero.

Now, we can simplify and generalize Definition 3.1 by redefining $\underline{d}_i$ and $\underline{q}$ in the following manner. Suppose $d_j^{(i)} = y_j$ for all $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$ with the y_j bounded on $[a, b]$, where $b > a > 0$ are real constants. Let $\underline{q} = (q_1, q_2, \ldots, q_m)$ such that all q_j, $j = 1, 2, \ldots, m$ are bounded on $[a', b']$ where $b' > a' > 0$ are real constants and b' and a' are some multiple of b and a respectively. Recall that the derivation of the significance values of terms is based on some statistical association measures which are in turn a function of frequencies of occurrence and co-occurrence of terms. Hence, the significance value of a term is a measure of its relative importance with others, and it is not a measure of absolute importance. As a result, the vectors $\underline{d}_i$ and $\underline{q}$ are

closely related if the angle between them is close to zero. Hence, we can assign (β_1, β_2) to be (1 ,0). Consequently, we can define a simplified and generalized relevance criterion for the present system as given in Definition 3.2.

Definition 3.2

A document $\underline{d}_i$ is said to be relevant to a given query $\underline{q}$ if and only if

$$T \leq R(\underline{d}_i, \underline{q}) \leq 1, \tag{3.6}$$

where $R(\underline{d}_i, \underline{q}) = \underline{d}_i \cdot \underline{q} / \| \underline{d}_i \| \| \underline{q} \|$, (3.7)

= relevance measure of $\underline{d}_i$ to $\underline{q}$,

and, T = a pre-determined cutoff value.

The correspondence between (3.6) and (3.4) is easy to derive. According to the above arguments, $(\beta_1, \beta_2) = (1, 0)$. Therefore, from (3.4), we have

$$1 - R(\underline{d}_i, \underline{q}) \leq \xi.$$

Then, $T = 1 - \xi \leq R(\underline{d}_i, \underline{q})$. Since $R(\underline{d}_i, \underline{q})$ is a measure of the cosine of the angle between $\underline{d}_i$ and $\underline{q}$, it is naturally less than or equal to one. The derivation of (3.6) is thus completed.

There are a few interesting properties associated with the relevance measure $R(\underline{d}_i, \underline{q})$. They are as follows:

(1) When $R(\underline{d}_i, \underline{q}) = 1$, condition one of Definition 3.1 is maintained.

(2) $\theta = \arccos\{R(\underline{d}_i, \underline{q})\}$ is the angular distance between $\underline{d}_i$ and $\underline{q}$, $|\theta| \leq \pi$.

(3) The function $\rho(\underline{d}_i, \underline{q}) = 1 - R(\underline{d}_i, \underline{q})$, in which $R(\underline{d}_i, \underline{q})$ satisfies the condition of (3.6), is a metric which satisfies the following axicms:

(i) $\rho(\underline{d}_i, \underline{q}) \geq 0$ and $\rho(\underline{d}_i, \underline{d}_i) = 0$.

(ii) $\rho(\underline{d}_i, \underline{q}) = \rho(\underline{q}, \underline{d}_i)$.

(iii) $\rho(\underline{d}_i, \underline{d}_j) \leq \rho(\underline{d}_i, \underline{q}) + \rho(\underline{q}, \underline{d}_j)$.

(iv) If $\underline{d}_i \neq \underline{q}$, then $\rho(\underline{d}_i, \underline{q}) > 0$.

(4) When $R(\underline{d}_i, \underline{q}) = 0$, $\underline{d}_i \cdot \underline{q} = 0$ implies that the two vectors do not have a single term in common.

Having defined the relevance measure, it is then possible to define the degree of relevance of a document in relation to a given query. This is given in Definition 3.3.

<u>Definition 3.3</u>

Let $D_R = \{\underline{d}_i,\ i = 1, 2, \ldots, k\}$ be the set of k documents judged relevant to a given query $\underline{q}$ by Definition 3.2. Suppose $R^* = \{r_i,\ i = 1, 2, \ldots, k\}$ is the corresponding set of relevance values determined by (3.7). Then the degree of

relevance of any document $\underline{d}_i \in D_R$ to $\underline{q}$ may be defined to be the relevance value r_i. Furthermore, a document $\underline{d}_i \in D_R$ is said to be more relevant to $\underline{q}$ than any other document $\underline{d}_j \in D_R$ to $\underline{q}$ if and only if the relevance value of $\underline{d}_i$ is greater than that of $\underline{d}_j$, i.e. $r_i > r_j$.

According to this definition, the set of provisionally relevant documents can be arranged in descending order of their relevance values thus showing the relative degree of relevance of the documents to the given query. This arrangement enables the system to decide which members of this set are indeed relevant. It also plays an important role in the query modification in the optimum itertaive feedback algorithm to follow.

CHAPTER IV

THE OPTIMUM ITERATIVE FEEDBACK ALGORITHM

4.1 Optimum Feedback Parameters

Before discussing the development of the optimum feedback algorithm, it is necessary to give a general description of the parameters involved. There are two interrelated sets of parameters. They are respectively called the set of user parameters and the set of system parameters.

It is observed from the formal query language definition given in section 3.1 that the user may specify the kind of output he anticipates in terms of recall and precision. Suppose $E(r)$ is the expected recall value and $E(p)$ is the expected precision value both belong to the interval $[0, 100]$. Now, let

$$r = \text{normalized expected recall value.}$$
$$= E(r)/100, \tag{4.1}$$

and,

$$p = \text{normalized expected precison value.}$$
$$= E(p)/100. \tag{4.2}$$

Suppose a set of $m' \leq m$ number of title terms are used in the query. Then the set of term weights assigned by the user $Q = \{q_i, i = 1, 2, \ldots, m'\}$, together with r and p form the set of user parameters U which is denoted in set notation as

$$U = \{r, p, Q\}. \tag{4.3}$$

Recall from Definition 3.2 that some document $\underline{d}_i$ is regarded as relevant to the query $\underline{q}$ if and only if $T \leq R(\underline{d}_i, \underline{q}) \leq 1$. Note that the system transforms the set of term weights, Q, into a vector $\underline{q} = (q_1, q_2, \ldots, q_m)$ by rearranging the subscripts according to the order of the index terms. For any $q \not\in Q$, the value of zero is inserted. From the system's point of view, the value of T should be bounded so that the validity of the relevance measure and the effectiveness of system performance are maintained. Since the closer the value of $R(\underline{d}_i, \underline{q})$ is to one, the higher is the degree of relevance of the retrieved document $\underline{d}_i$. Hence, it is natural to think that T should be assigned as close to one as possible. However, having T too close to one will most likely result in high precision but low recall. Conversely, having T too far away from one will most likely result in high recall and low precision. In order to compromise this, let us define a threshold value T in terms of r and p as

$$T = \max\{0.7,\ 1 - |\log r/(1+r^2)| - |\log p/(1+p^2)|\}, \tag{4.4}$$

where log is the common logarithm. Alternatively, (4.4) may be represented approximately by

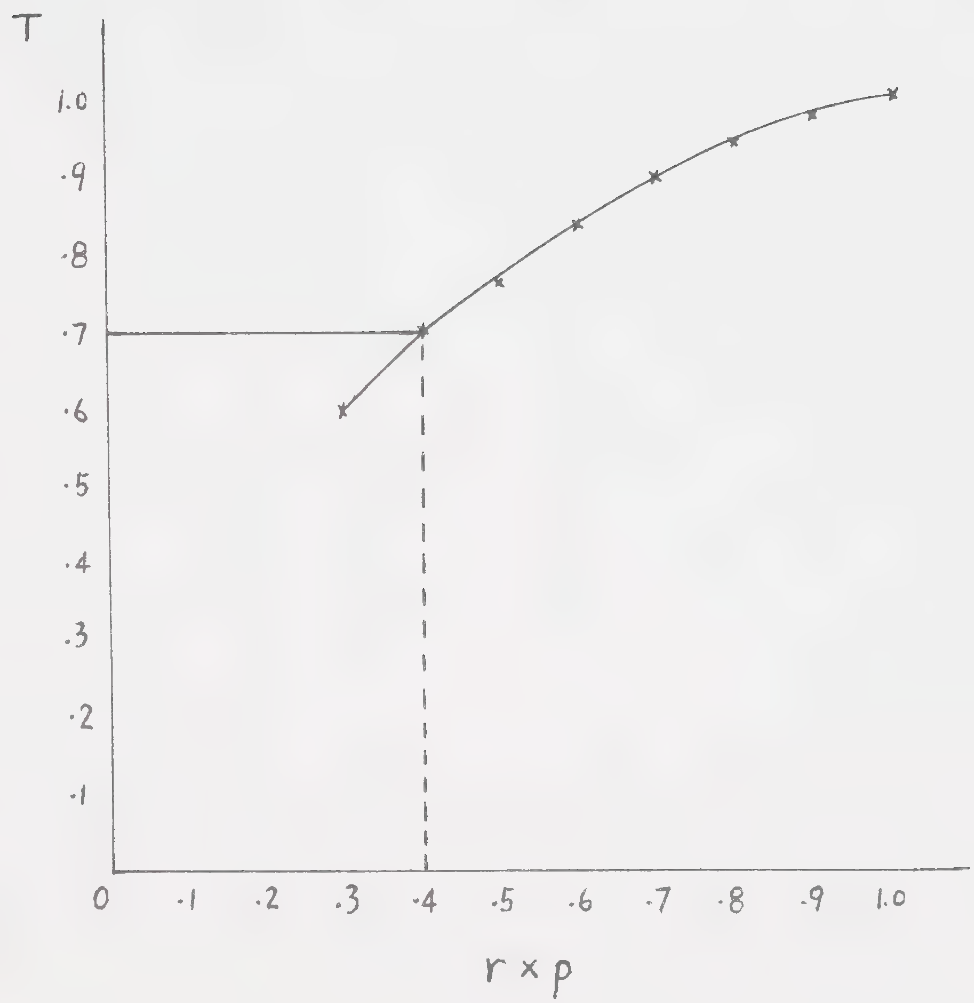
$$T \simeq \begin{cases} 1 - |\log r/(1+r^2)| - |\log p/(1+p^2)|, \text{ if } r.p \geq .4. & (4.5a) \\ .7, \text{ if } r.p < .4. & (4.5b) \end{cases}$$

The effect of (4.4) results in bounding T in the interval [0.7, 1.0], which is a reasonable range to maintain effective selection of relevant documents. The values of T as given by (4.5a) for the values of r and p between 0.5 and 1.0 in steps of 0.1 are given in Table 4.1. The graphs of T versus r.p as defined by (4.5b) are given in Fig. 4.1.

r \ T \ p	0.5	0.6	0.7	0.8	0.9	1.0
0.5	0.518	0.596	0.655	0.700	0.734	0.759
0.6	0.596	0.674	0.733	0.778	0.812	0.837
0.7	0.655	0.733	0.792	0.837	0.871	0.896
0.8	0.700	0.778	0.837	0.882	0.916	0.941
0.9	0.734	0.812	0.871	0.916	0.949	0.975
1.0	0.759	0.837	0.896	0.941	0.975	1.000

Table 4.1 _Values of T for Given r and p_

Fig. 4.1 Graphs of T vs r×p



Consider the case when no document is judged relevant to a request by the relevance criterion of (3.6). Suppose there are some documents whose relevance values are short of the value of T but are, to a certain extent, close to it. The possible source of discrepancy may come from the set of term weights, Q, assigned by the user. Then, it is quite possible that by modifying the original search request, some of these documents or some other related documents in the data base may be judged as relevant to the request. It is therefore necessary to define another threshold value T' in which $T > T'$, so that any provisionally relevant document whose relevance value falls into the interval $[T', T]$ may be considered as capable of being improved. The expression for T' may be defined as:

$$T' = 2T - 1, \tag{4.6a}$$

$$\quad = \max\{0.4,\ 1 - 2|\log r/(1+r^2)| - 2|\log p/(1+p^2)|\}. \tag{4.6b}$$

As before, T' may be represented approximately by the following relations:

$$T' \simeq \begin{cases} 1 - 2|\log r/(1+r^2)| - 2|\log p/(1+p^2)|, \text{ if } r.p \geq .4. & (4.7a) \\ .4, \text{ if } r.p < .4. & (4.7b) \end{cases}$$

The effect of (4.6) results in bounding T' in the interval [0.4, 1.0]. The values of T' as given by (4.7a) for the values of r and p between 0.5 and 1.0 in steps of 0.1 are given in Table 4.2. The graphs of T' versus r.p as defined by (4.7a) and

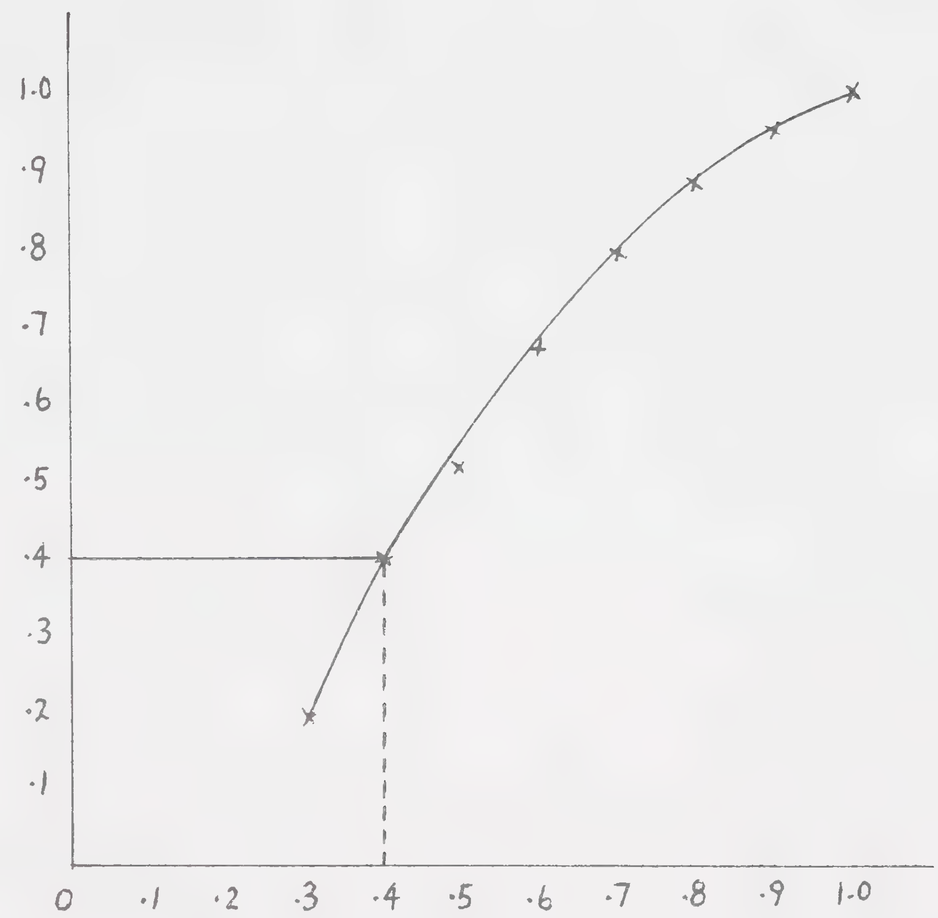
(4.7b) are given in Fig. 4.2.

r \ T' \ p	0.5	0.6	0.7	0.8	0.9	1.0
0.5	0.037	0.192	0.310	0.400	0.468	0.518
0.6	0.192	0.348	0.466	0.556	0.623	0.674
0.7	0.310	0.466	0.584	0.674	0.742	0.792
0.8	0.400	0.556	0.674	0.764	0.831	0.882
0.9	0.468	0.623	0.742	0.831	0.899	0.950
1.0	0.518	0.674	0.792	0.882	0.950	1.000

Table 4.2 Values of T' for Given r and p

It is worthwhile to note that the definitions of the threshold values as given by (4.4) and (4.6a) are obtained empirically. A system operator is therefore free to modify these values according to need. Alternatively, the two threshold values may appear in the form of parameters to be supplied by the users. In the present system we define the cutoff value τ to be in the interval [T', T]. Then, according to Definition 3.2, a document is relevant to a given query if and only if its relevance value lies in the closed interval [τ, 1].

Fig. 4.2 Graphs of T' vs r×p



Consequently, a query $\underline{q}$ is considered to be capable of being improved if there exists at least one document $\underline{d}_i$ such that

$$T' \leq R(\underline{d}_i, \underline{q}) < T. \tag{4.8}$$

The set of significance values of index terms $Y = \{y_j, j = 1, 2, \ldots, m\}$ obtained by the automatic indexing algorithm, together with T, T and T' form the set of system parameters S which is denoted in set notation as

$$S = \{ T, T, T', Y\}. \tag{4.9}$$

4.2 The Optimum Iterative Feedback Algorithm

In document retrieval systems, the optimization of retrieval output is to find a set of documents which satisfy some pre-determined criteria utilizing some known parameters. In the present system, the two sets of parameters U and S play an important role in the optimization process. Consider a given query $\underline{q} = (q_1, q_2, \ldots, q_m)$. Let $D_R = \{\underline{d}_i, i = 1, 2, \ldots, h\}$ be the set of h documents which satisfy the relevance criterion $T \leq R(\underline{d}_i, \underline{q}) \leq 1$. Similarly, let $D'_R = \{\underline{d}_j, j = 1, 2, \ldots, h'\}$ be the set of h' documents which satisfy the condition of (4.8).

Let ϕ represent a null set. Suppose $D_R \neq \phi$, then the set of documents that satisfy condition (3.6) will be considered as

relevant hits, and presented to the user in descending order of their degree of relevance. Now, suppose $D_R = \phi$ and $D'_R \neq \phi$, it is then possible to modify the original (or previously modified) search request and pass control back to the search phase to re-examine if indeed any of the documents $\varepsilon D'_R$ may now be judged as relevant hits by the relevance criterion of (3.6). Define the most satisfactory document $\underline{d}_\ell \varepsilon D'_R$ for improvement as the one such that

$$R(\underline{d}_\ell, \underline{q}) = \max_{\underline{d}_i \varepsilon D'_R} \{R(\underline{d}_i, \underline{q})\}. \tag{4.10}$$

A set of new symbols are introduced to facilitate the explanation of the algorithm. Let $\underline{q}^{(k)}$ be the query in conjunction to the k-th iteration, for some k = 0, 1, 2, ... , such that $\underline{q}^{(0)} = \underline{q}$ and $\underline{q}^{(k)} = (q_1^{(k)}, q_2^{(k)}, \ldots, q_m^{(k)})$. Suppose $D_R^{(k)} = \{\underline{d}_i^{(k)}, i = 1, 2, \ldots, h^{(k)}\}$ is the set of $h^{(k)}$ documents judged relevant to $\underline{q}$ by the relevance criterion $T \leq R(\underline{d}_i^{(k)}, \underline{q}^{(k)}) \leq 1$ in conjunction to the k-th iteration. Similarly, suppose $D_R'^{(k)} = \{\underline{d}_j^{(k)}, j = 1, 2, \ldots, h'^{(k)}\}$ is the set of $h'^{(k)}$ documents judged relevant to $\underline{q}$ by the condition of (4.8) in conjunction to the k-th iteration. Also, $D_R^{(0)} = D_R$ and $D_R'^{(0)} = D'_R$. Suppose $D_R^{(k)} = \phi$ and $D_R'^{(k)} \neq \phi$ for some k, then the (k+1)-st modified query $\underline{q}^{(k+1)}$ can be defined as:

$$\underline{q}^{(k+1)} = \underline{q}^{(k)} + \alpha_k \underline{\Omega}^{(k)}, \tag{4.11}$$

in which,

$$\alpha_k = \underline{d}_\ell^{(k)} \cdot \underline{g}^{(k)} / \| \underline{d}_\ell^{(k)} \|^2, \tag{4.12}$$

$$\Omega_i^{(k)} = \lambda_i^{(k)} d_i^{(k)}, \tag{4.13}$$

$$\lambda_i^{(k)} = \begin{cases} +1, & \text{if } q_i^{(k)} < d_i^{(k)}, & (4.14a) \\ -1, & \text{if } q_i^{(k)} > d_i^{(k)}, & (4.14b) \\ 0, & \text{if } q_i^{(k)} = d_i^{(k)}, & (4.14c) \end{cases}$$

such that $d_i^{(k)}$ is the i-th element of $\underline{d}_\ell^{(k)}$, and $q_i^{(k)}$ is the i-th element of $\underline{g}^{(k)}$, $\Omega_i^{(k)} \varepsilon \underline{\Omega}^{(k)}$, $i = 1, 2, \ldots, m$; and, $\underline{d}_\ell^{(k)} \varepsilon D_R'^{(k)}$ such that,

$$R(\underline{d}_\ell^{(k)}, \underline{g}^{(k)}) = \max_{\underline{d}_i^{(k)} \varepsilon D_R'^{(k)}} \{R(\underline{d}_i^{(k)}, \underline{g}^{(k)})\}. \tag{4.15}$$

If $D_R^{(k)} = \phi$ and $D_R'^{(k)} = \phi$ for some k, then the query $\underline{q}$ will be considered as having no hits. The sequence of queries $\{\underline{g}^{(0)}, \underline{g}^{(1)}, \ldots, \underline{g}^{(k)}, \ldots\}$ performs the necessary iterative control over the decision of an optimum set of retrieved documents. This sequence can be proved to be convergent and the proof is given in the next section. In summary, the optimum iterative feedback algorithm is given as follows:

<u>STEP</u> 1: Set $k = 0$. Receive $\underline{q}^{(o)} = (q_1^{(o)}, q_2^{(o)}, \ldots, q_m^{(o)})$ and scale each $q_i^{(o)}$, $i = 1, 2, \ldots, m$ in the interval defined by $[\min_j y_j, \max_j y_j]$.

<u>STEP</u> 2: Determine T and T'.

<u>STEP</u> 3: Retrieve $\underline{d}_j^{(k)}$, $j = 1, 2, \ldots, n$; at end, go to Step 5.

<u>STEP</u> 4: (i) If $T \leq R(\underline{d}_i^{(k)}, \underline{q}^{(k)}) \leq 1$, form $D_R^{(k)}$. Go to Step 3.

(ii) If $T' \leq R(\underline{d}_i^{(k)}, \underline{q}^{(k)}) < T$, form $D_R'^{(k)}$. Go to Step 3.

(iii) Otherwise, go to Step 3.

<u>STEP</u> 5: (i) If $D_R^{(k)} \neq \phi$, go to Step 6.

(ii) If $D_R^{(k)} = \phi$ and $D_R'^{(k)} = \phi$, go to Step 7.

(iii) Otherwise, determine $\underline{d}_\ell^{(k)}$ such that $\underline{d}_\ell^{(k)} \varepsilon D_R'^{(k)}$ and $R(\underline{d}_\ell^{(k)}, \underline{q}^{(k)}) = \max_{\underline{d}_i^{(k)} \varepsilon D_R'^{(k)}} \{R(\underline{d}_i^{(k)}, \underline{q}^{(k)})\}$, and set:

$$\underline{q}^{(k+1)} = \underline{q}^{(k)} + \alpha_k \underline{\Omega}^{(k)},$$

where α_k is as defined by (4.12) and $\underline{\Omega}^{(k)}$ is defined by (4.13); and $\lambda_i^{(k)}$ is defined by (4.14a), (4.14b) and (4.14c); $d_i^{(k)} \varepsilon \underline{d}_\ell^{(k)}$, $i = 1, 2, \ldots, m$. Then increment k by one and go to Step 3.

<u>STEP</u> 6: Arrange $R(\underline{d}_i^{(k)}, \underline{q}^{(k)})$, for all $\underline{d}_i^{(k)}$ such that $T \leq R(\underline{d}_i^{(k)}, \underline{q}^{(k)}) \leq 1$, into descending order of relevance value

and output documents as relevant hits. Then go to Step 8.

STEP 7: Display "no hits" message.

STEP 8: STOP.

Finally, the flowchart for the optimum iterative feedback algorithm is given in Fig. 4.3.

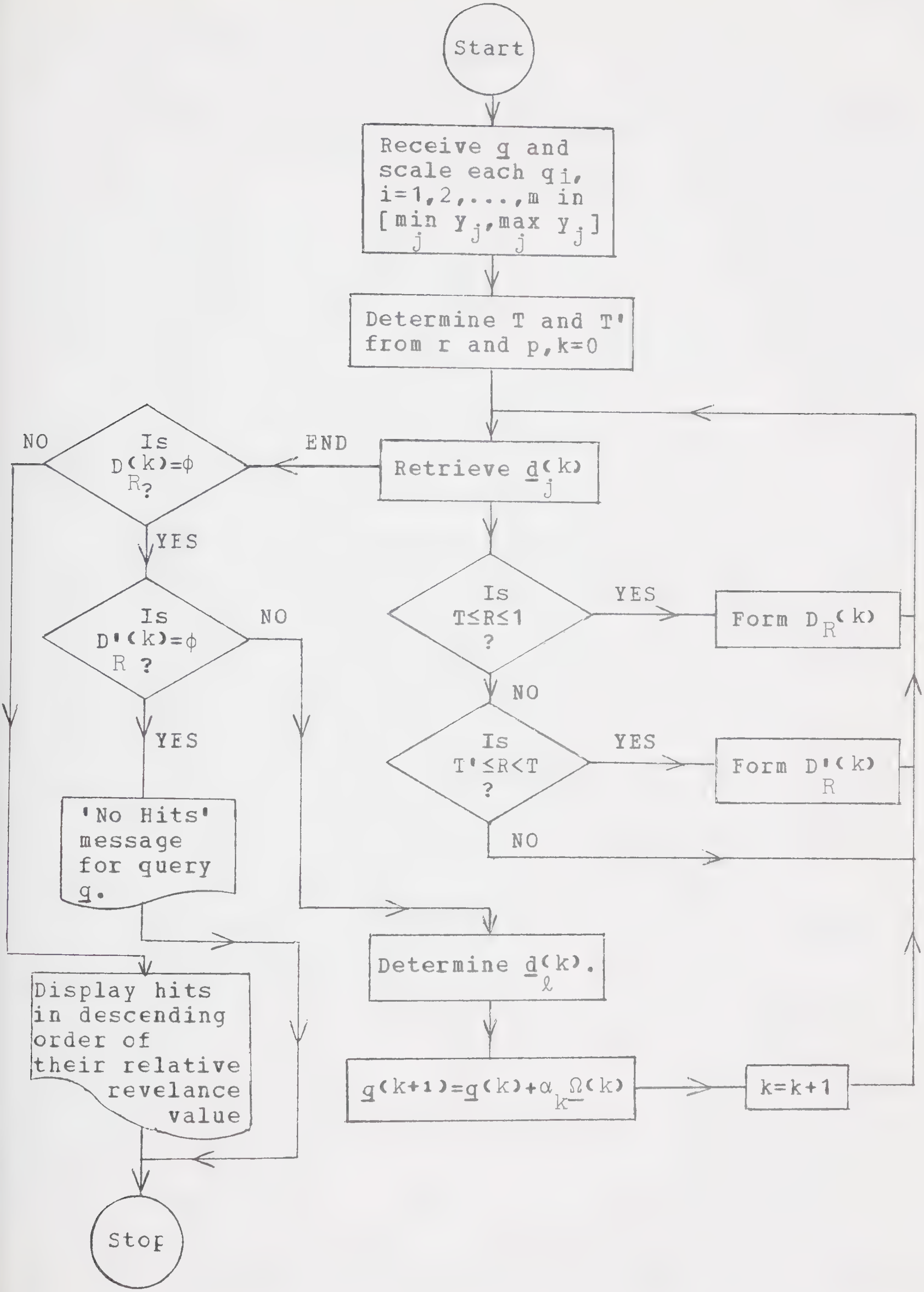


Fig.4.3 Flowchart for the Optimum Iterative Feedback Algorithm

4.3 Convergence of the Algorithm

The above algorithm is convergent if there exists a document $\underline{d}^* \varepsilon D$ such that after n finite number of iterations, $T \leq R(\underline{d}^*, \underline{g}^{(n)}) \leq 1$. In other words, if the set $D_R^{(n)}$ becomes non-empty after n iterations, n must have a lower bound and an upper bound. The proof is trivial and is given as follows:

Proof

Suppose after n iterations, there exists a document $\underline{d}^* \varepsilon D$ such that $T \leq R(\underline{d}^*, \underline{g}^{(n)}) \leq 1$. Then, by definition,

$$R(\underline{d}^*, \underline{g}^{(n)}) = \underline{d}^*.\underline{g}^{(n)}/|| \underline{d}^* || \; || \underline{g}^{(n)} ||. \qquad (4.16)$$

We have, from (4.11),

$$\underline{d}_\ell^{(k)}.\underline{g}^{(k+1)} = \underline{d}_\ell^{(k)}.\underline{g}^{(k)} + \alpha_k (\underline{d}_\ell^{(k)}.\underline{\Omega}^{(k)}). \qquad (4.17)$$

From (4.13), it is obvious that,

$$\underline{\Omega}^{(k)} \leq \underline{d}_\ell^{(k)}. \qquad (4.18)$$

Then (4.17) becomes

$$\underline{d}_\ell^{(k)}.\underline{g}^{(k+1)} \geq \underline{d}_\ell^{(k)}.\underline{g}^{(k)} + \alpha_k || \underline{\Omega}^{(k)} ||^2. \qquad (4.19)$$

Suppose $\omega = \min_k \| \underline{\Omega}^{(k)} \|$. Then, after n iterations,

$$\underline{d}^{*}.\underline{g}^{(n)} \geq n\alpha_n\omega^2. \tag{4.20}$$

Similarly, we have

$$\|\underline{g}^{(k+1)}\|^2 \leq \| \underline{g}^{(k)} \|^2 + 2\alpha_k(\underline{d}_\ell^{(k)}.\underline{g}^{(k)}) + \alpha_k{}^2\| \underline{d}_\ell^{(k)} \|^2. \tag{4.21}$$

$$= \| \underline{g}^{(k)} \|^2 + 3(\underline{d}_\ell^{(k)}.\underline{g}^{(k)}/\| \underline{d}_\ell^{(k)} \|)^2. \tag{4.22}$$

Suppose $\delta = \max_k \| \underline{d}_\ell^{(k)} \|$. Then, after n iterations,

$$\| \underline{g}^{(n)} \| \leq \sqrt{3n}\alpha_n\delta. \tag{4.23}$$

Finally, combining (4.20) and (4.23), we obtain,

$$3T^2\sigma \leq n \leq 3\sigma, \tag{4.24}$$

where $\sigma = \delta^2/\omega^4$. Since δ and ω are finite, therefore n is bounded.

(Q.E.D.)

4.4 Generalization of the Algorithm

It is noted that the algorithm described above takes into account of document title terms only. In general, it is desirable to include search on other items such as author names, journal coden names, year of publication and so forth. Suppose there are t different search items other than document title terms, we may let the modified (or augmented) document vector $\underline{d}_i'$ be such that

$$\underline{d}_i' = (\underline{d}_i : \underline{a}_1 : \underline{a}_2 : \ldots : \underline{a}_t), \tag{4.25}$$

and the modified (or augmented) query vector $\underline{g}'$ be such that

$$\underline{g}' = (\underline{g} : \underline{b}_1 : \underline{b}_2 : \ldots : \underline{b}_t), \tag{4.26}$$

where $\underline{a}_h$ and $\underline{b}_h$, h = 1, 2, ... , t are the subvectors associated with each search item; $\underline{d}_i$ and $\underline{g}$ being the usual document vector and query vector respectively. Furthermore, each $\underline{a}_h$ and $\underline{b}_h$ has the same dimension so that $\underline{d}_i'$ and $\underline{g}'$ are also of equal dimensions.

Without further knowledge of how much weight should be assigned to the elements of the subvectors $\underline{a}_h$ and $\underline{b}_h$, h = 1, 2, ... , t, we may assign a value to the j-th element of the h-th subvector denoted by $a_h^{(j)}$ such that

$$a_h^{(j)} = \begin{cases} 1, & \text{if } a_h^{(j)} \in \underline{d}_i'. & (4.27\text{a}) \\ 0, & \text{otherwise.} & (4.27\text{b}) \end{cases}$$

Similarly, we may assign a value to the j-th element of the h-th subvector denoted by $b_h^{(j)}$ such that

$$b_h^{(j)} = \begin{cases} 1, & \text{if } b_h^{(j)} \varepsilon \underline{g}'. \qquad (4.28a) \\ 0, & \text{otherwise.} \qquad (4.28b) \end{cases}$$

We may further assume that the values of $a_h^{(j)}$ and $b_h^{(j)}$, for all h and j, are invariate under query modifications. In other words, query modifications apply only to the subvector $\underline{g}$ in the new (or generalized) system, where all document vecotrs and query vectors are replaced by $\underline{d}_i'$ and $\underline{g}'$ respectively. The (k+1)-st modified query $\underline{g}'^{(k+1)}$ is therefore symbolically represented by

$$\underline{g}'^{(k+1)} = (\underline{g}^{(k)} + \alpha_k \underline{\Omega}^{(k)} : \underline{b}_1 : \underline{b}_2 : \ldots : \underline{b}_t). \qquad (4.29)$$

Consequently, the generalized system can be proved to be convergent in that

$$T^2\sigma' \leq n \leq \sigma', \qquad (4.30)$$

where $\sigma' = (3\alpha_n{}^2\delta^2 + \gamma)/(\alpha_n\omega^2 + \beta)^2$; $\gamma = \sum_{i=1}^{t} (\underline{a}_i \cdot \underline{b}_i)$; and $\beta = \sum_{i=1}^{t} (\underline{b}_i \cdot \underline{b}_i)$.

CHAPTER V

GENERAL DISCUSSIONS

5.1 A System Evaluation Measure

Conventionally, the standard evaluation measure of system effectiveness is defined in terms of recall and precision. Recall is the proportion of relevant documents actually retrieved; while precision is the proportion of retrieved documents actually relevant. Let

N = the total number of documents in D,

x = the number of documents retrieved and relevant,

y = the number of documents retrieved and not relevant, and

z = the number of documents not retrieved and relevant.

Then the number of documents not retrieved and not relevant is N-x-y-z. We may construct a two-by-two contingency table as shown in Table 5.1 to represent the above situation.

	Retrieved	Not-Retrieved	Total
Relevant	x	z	x+z
Not-Relevant	y	N-x-y-z	N-x-z
Total	x+y	N-x-y	N

Table 5.1 2-by-2 Contingency Table of Retrieval and Relevance

Then, by definition,

$$\text{Recall} = x/(x+z), \tag{5.1}$$

and,

$$\text{Precision} = x/(x+y). \tag{5.2}$$

It can be observed that for systems in which retrieval is not based on relevance, the determination of recall and precision values according to (5.1) and (5.2) requires a tremendous amount of manual work. While this method of system evaluation may be all right for a small data base, it is absolutely impractical in normal situations where very large

collections of data are involved. Consequently, system evaluation in terms of recall and precision may be treated as a theoretical entity.

As mentioned earlier, the prime objective of a document retrieval system is to achieve the retrieval of all relevant, and only relevant, documents in response to any user's query. Hence, we would want to define a system evaluation measure in terms of relevance instead of recall and precision. Note that in either case, the meaning of relevance must be well-defined. In the present system, the meaning of relevance is given in Definition 3.2. Now, let

D' = {Set of documents retrieved}, and

D'' = {Set of documents not retrieved}.

Then, by definition,

$$D = D' \cup D''. \tag{5.3}$$

where $\cup$ represents the union of sets.

Suppose, as usual, $\underline{q}$ represents a query vector, and $\underline{d}_i$, $i = 1, 2, \ldots, N$ represents a document vector in D. It can be pointed out from Table 5.1 that there can be four cases:

I. $\underline{d}_i \in D'$ and relevant;

II. $\underline{d}_i \in D'$ and not relevant;

III. $\underline{d}_i \in D''$ and relevant; and

IV. $\underline{d}_i \in D''$ and not relevant.

Obviously, we would like to have all $\underline{d}_i$ retrieved in response to $\underline{q}$ to satisfy only case I <u>and</u> all $\underline{d}_i$ not retrieved in response to $\underline{q}$ to satisfy only case IV. In general, we may define a system evaluation measure E to be

$$E = \delta_1 \sum_{\underline{d}_i \varepsilon I} R(\underline{d}_i,\underline{q}) - \delta_2 \sum_{\underline{d}_i \varepsilon II} R(\underline{d}_i,\underline{q})$$

$$- \delta_3 \sum_{\underline{d}_i \varepsilon III} R(\underline{d}_i,\underline{q}) + \delta_4 \sum_{\underline{d}_i \varepsilon IV} R(\underline{d}_i,\underline{q}), \qquad (5.4)$$

where $\delta_1 = 1/x,$ (5.5a)

$\delta_2 = 1/y,$ (5.5b)

$\delta_3 = 1/z,$ (5.5c)

$\delta_4 = 1/(N-x-y-z).$ (5.5d)

Note that the value of z is an unknown. However, there are different techniques such as statistical sampling methods that may be applied to determine z. Since relevance values are actually part of the calculations of the search process in the present system, the evaluation of E is quite simple and straightforward. Hence, it can be concluded that the definition of E in terms of relevance is far more practical than in terms of recall and precision.

In any given document retrieval system, system performance is considered to be optimum if the value of a given system evaluation measure is a minimum or a maximum. For example, a system whose performance is based on recall and precision will

require either the recall or the precision be a maximum in order to achieve optimum system performance. According to the definition of E, to optimize the system performance of any given system would mean the minimization of E. As for the present iterative feedback system, it can be observed that the value of E tends to be a minimum and therefore the set of retrieved documents can be regarded as optimum in terms of system performance.

5.2 Analysis of Search Output

The optimum iterative feedback algorithm and the generalized optimum iterative feedback algorithm are tested on an IBM 360/67 under the Michigan Terminal System (MTS). The programming language used is FORTRAN IV level G. In the paragraphs to follow, two sample search requests are given to illustrate the performance of the algorithms. Searches are performed on the first four hundred documents of CSDATA.

(I) The Optimum Iterative Feedback Algorithm

Sample search request one is given as follows:

```
QUE                                                    60  70
   TEST OF THE OPTIMUM ITERATIVE FEEDBACK ALGORITHM
AND T DOCUMENT                                            800
 OR T INFORMATION                                         500
 OR T STORAGE                                             600
 OR T RETRIEVAL                                           400
END
```

The threshold values are calculated to be (T', T) = (0.466, 0.733) and the request vector after normalization becomes (0.674, 0.421, 0.505, 0.337) which corresponds to the terms in brackets as (document, information, storage, retrieval). The following list of documents and their relevance values is a result of the search process:

Relevance Value	Document
0.715	AMDOA68190071VENEZ/STORA RETRI EDITI INFOR DICTI
0.677	AMDOA68190381OCONN/RETRI ANSWE PROVI DOCUM
0.674	AMDOA67180249DRABH/DOCUM THAIL
0.674	AMDOA66170141SAVAG/USERS VERSU DOCUM
0.657	AMDOA68190173STARK/WHALE/CARSC/THOMP/GAF DOCUM STORA RETRI SYSTE
0.510	AMDOA65160005DALE /DALE /CLUMP EXPER ASSOC DOCUM RETRI
0.505	AMDOA68190363BAKER/NANCE/USE SIMUL STUDY INFOR STORA RETRI SYSTE
0.501	AMDOA64150150KENT /HEURI INFOR RETRI GAME
0.491	AMDOA69200311OCONN/INDEP AGREE RESOL DISAG ANSWE PROVI DOCUM
0.486	AMDOA67180010LIBBE/USE SECON ORDER DESCR DOCUM RETRI
0.475	AMDOA70210237KEITH/GENER EVALU INFOR STORA RETRI SYSTE

Since the highest relevance value of this set of documents is 0.715 which is greater than 0.466 but less than 0.733, query

modification will take place. The new request vector now becomes (0.674, 0.882, 0.851, 0.761, 0.0, 0.0) which corresponds to the terms in brackets as (document, information, storage, retrieval, editing, dictionary). As the terms 'editing' and 'dictionary' are insignificant in CSDATA, their significance values are negligible. The value of α_1 is found to be 0.516. Consequently, the output of the second search is given as follows:

Relevance Value	Document
0.898	AMDOA68190071VENEZ/STORA RETRI EDITI INFOR DICTI
0.643	AMDOA68190173STARK/WHALE/CARSO/THOMP/GAF DOCUM STORA RETRI SYSTE
0.637	AMDOA68190381OCONN/RETRI ANSWE PROVI DOCUM
0.634	AMDOA68190363BAKER/NANCE/USE SIMUL STUDY INFOR STORA RETRI SYSTE
0.631	AMDOA65160163HEILP/GOODM/ANALO BETWE INFOR RETRI EDUCA
0.596	AMDOA70210237KEITH/GENER EVALU INFOR STORA RETRI SYSTE
0.554	AMDOA70210089OTTEN/DEBON/METAS INFOR
0.549	AMDOA64150210KLEMP/METHO COMPA ANALY INFOR STORA RETRI SYSTE CRITI REVIE
0.542	AMDOA69200072SWETS/EFFEC INFOR RETRI METHO
0.521	AMDOA68190090SMITH/LEVY /PHYSI ORIEN METHO CLINI INFOR RETRI
0.517	AMDOA68190387POLLO/MEASU COMPA INFOR RETRI SYSTE
0.513	AMDOA70210154WININ/DATA STRUC INFOR RETRI
0.509	AMDOA70210004HUMPH/INFOR PEACE
0.483	AMDOA65140014VERHO/BELZE/USE META LANGU UNFOR RETRI

SYSTE

0.480 AMDOA68190275COTTR/EVALU COMPU SCIEN TECHN INFOR NUCLE SAFET INFOR CENTE

0.479 AMDOA65160005DALE /DALE /CLUMP EXPTA ASSOC DOCUM RETRI

0.478 AMDOA68190404TREU /BROWS RETRI GAME

0.478 AMDOA65160291GARVI/INFOR SURVE MODER LINGU

0.474 AMDOA68190200OCONN/QUEST CONCE INFOR NEED

The highest value of relevance of this search is [illegible], which is greater than 0.733. Therefore, this set of documents will be regarded as the final search output to the given search request using a cutoff value of 0.466. It is noted that the documents in this set are all related one way or another to the subject of document/information storage and retrieval. In practice, if the estimated recall and precision values are varied, the number of hits will also be changed. For instance, if a higher demand for recall and precision is imposed, fewer number of documents will likely be considered as relevant hits.

It is important to realize that the search process depends a great deal on the term weights assigned by the user. Suppose, in the above search request, the weights are changed to (892, 822, 669, 630) in correspondence with the terms in barckets as (document, information, storage, retrieval). These weights are obtained from the set of index terms and their significance values. By using the same estimated recall and precision values, it is found that no iterative search is required. The search output is given as follows:

Relevance Value	Document
0.910	AMDOA68190071VENEZ/STORA RETRI EDITI INFOR DICTI
0.742	AMDOA64150150KENT /HEURI INFOR RETRI GAME
0.688	AMDOA65160163HEILP/GOODM/ANALO BETWE INFOR RETRI EDUCA
0.681	AMDOA68190381OCONN/RETRI ANSWE PROVI DOCUM
0.643	AMDOA68190363BAKER/NANCE/USE SIMUL STUDY INFOR STORA RETRI SYSTE
0.631	AMDOA68190173STARK/WHALE/CARSO/THOMP/GAF DOCUM STORA RETRI SYSTE
0.604	AMDOA70210237KEITH/GENER EVALU INFOR STORA RETRI SYSTE
0.591	AMDOA69200072SWETS/EFFEC INFOR RETRI METHO
0.586	AMDOA68190286MILLE/PSYCH INFOR
0.586	AMDOA70210089OTTEN/DEBON/METAS INFOR
0.568	AMDOA68190090SMITH/LEVY /PHYSI ORIEN METHO CLINI INFOR RETRI
0.564	AMDOA68190387POLLO/MEASU COMPA INFOR RETRI SYSTE
0.559	AMDOA70210145WININ/DATA STRUC INFOR RETRI
0.556	AMDOA64150210KLEMP/METHO COMPA ANALY INFOR STORA RETRI SYSTE CRITI REVIE
0.540	AMDOA68190404TREU /BROWS RETRI GAME
0.538	AMDOA70210004HUMPH/INFOR PEACE
0.527	AMDOA64150014VERHO/BELZE/USE META LANGU INFOR RETRI SYSTE
0.512	AMDOA65160005DALE /DALE /CLUMP EXPER ASSOC DOCUM RETRI
0.509	AMDOA68190375COTTR/EVALU COMPR SCIEN TECHN INFOR NUCLE SAFET INFOR CENTE
0.506	AMDOA65160291GARVI/INFOR SURVE MODER LINGU

0.503	AMDOA67180235BUCHA/HUTTO/ANALY	AUTOM HANDL TECHN INFOR NUCLE SAFET INFOR
0.502	AMDOA68190200CONN/QUEST	CONCE INFOR NEED
0.492	AMDOA70210095BROMB/ECONO	INFOR
0.488	AMDOA67180010LIBBE/USE	SECON ORDER DESCR DOCUM RETRI
0.484	AMDOA70210385COOPE/DERIV	DESIG EQUAT INFOR RETRI SYSTE
0.479	AMDOA69200169FLANI/OPEN	ENDED INFOR RETRI SYSTE INCLU SELEC DATA COLLE
0.470	AMDOA69200039LUNIN/ACADE	INFOR CENTE
0.468	AMDOA68190305THOMP/ORGAN	INFOR

It is interesting to note that the previous set of search output is a subset of this set of search output which is heavily dependent on the system parameters and therefore may contain some information unexpected by the user. Therefore, in order that the user will receive the most satisfactory search output, he needs to make a good judgment of the use of term weights.

(II) The Generalized Optimum Iterative Feedback Algorithm

By modifying sample search request one we obtain sample search request two which is given as follows:

```
QUE                                                        60  70
    TEST OF THE GENERALIZED OPTIMUM ITERATIVE FEEDBACK ALGORITHM
AND A BAKER
 OR C AMDOA
 OR Y 68
 OR T DOCUMENT                                                800
 OR T INFORMATION                                             500
 OR T STORAGE                                                 600
 OR T RETRIEVAL                                               400
END
```

The threshold values and the weights assigned to terms are unchanged. It is found that only one document has relevance value above the threshold value 0.466 and is given as follows:

Relevance Value		Document
0.534	AMDOA68190363BAKER/NANCE/USE	SIMUL STUDY INFOR STORA RETRI SYSTE

As before, since the relevance value of the retrieved document lies in the interval [0.466, 0.733], the original query vector is modified. The new query vector includes the terms enclosed in brackets as (information, retrieval, storage, document, use, simulation, study, system) with the corresponding new set of weights as (0.534, 0.441, 0.590, 0.674, 0.0, 0.767, 0.598, 0.992). Since the term 'use' is insignificant in CSDATA,

its significance value is zero. The value of α_1 is found to be 0.126. The following list of documents and their relevance values are the result of the second search:

Relevance Value		Document
0.739	AMDOA68190363BAKER/NANCE/USE	SIMUL STUDY INFOR STORA RETRI SYSTE
0.503	AMDOA69200203BAKER/OPTIM USER	SEARC SEQUE IMPLI INFOR SYSTE OPERA
0.483	AMDOA69200027LESK /WORD WORD	ASSOC DOCUM RETRI SYSTE
0.483	AMDOA70210330CAGAN/HIGHL ASSCC	DOCUM RETRI SYSTE
0.477	AMDOA67180216FLOOD/ANALY QUEST	ASKED MEDIC REFER RETRI SYSTE COMPA QUEST SYSTE TERMI
0.476	AMDOA70210237KEITH/GENER EVALU	INFOR STORA RETRI SYSTE
0.474	AMDOA68190120CARAS/COMPU SIMUL	SMALL INFOR SYSTE
0.469	AMDOA67180055BAKER/HAEFE/RECKH/FILM	SYSTE DUPLI TERMA CARDS

The highest relevance value of this search is 0.739 which is greater than the threshold value 0.733. Therefore, this set of documents is regarded as the final search output to the given sample search request.

Sample search request two is now modified so that the weights corresponding to (document, information, storage, retrieval) become (892, 822, 669, 630). As before, these weights are obtained from the set of index terms and their significance

values. Unlike the case as given in (I), two searches are required to bring forth an optimum set of search output. The first set of documents is given as follows:

Relevance Value	Document
0.582	AMDOA68190363BAKER/NANCE/USE SIMUL STUDY INFOR STORA RETRI SYSTE
0.510	AMDOA68190071VENEZ/STORA RETRI EDITI INFOR DICTI

After query modification, the new request vector now includes the terms shown in brackets as (information, retrieval, storage, document, use, simulation, study, system) which corresponds to the weights given as (0.729, 0.672, 0.547, 0.414, 0.0, 0.767, 0.598, 0.992). The value of α_1 is found to be 0.161. Consequently, the final search output is:

Relevance Value	Document
0.777	AMDOA68190363BAKER/NANCE/USE SIMUL STUDY INFOR STORA RETRI SYSTE
0.521	AMDOA69200203BAKER/OPTIM USER SEARC SEQUE IMPLI INFOR SYSTE OPERA
0.519	AMDOA70210237KEITH/GENER EVALU INFOR STORA RETRI SYSTE
0.517	AMDOA68190387POLLO/MEASU COMPA INFOR RETRI SYSTE
0.498	AMDOA67180216FLOOD/ANALY QUEST ASKED MEDIC REFER RETRI SYSTE COMPA QUEST SYSTE TERMI
0.494	AMDOA68190120CARAS/COMPU SIMUL SMALL INFOR SYSTE

0.493	AMDOA64150210KLEMP/METHO	COMPA ANALY INFOR STORA RETRI SYSTE CRITI REVIE
0.481	AMDOA69200027LESK /WORD	WORD ASSOC DOCUM RETRI SYSTE
0.481	AMDOA70210330CAGAN/HIGHL	ASSOC DOCUM RETRI SYSTE
0.477	AMDOA66170026PARKE/USERS	PLACE INFOR SYSTE
0.477	AMDOA70210385COOPE/DERIV	DESIG EQUAT INFOR RETRI SYSTE
0.474	AMDOA69200169FLANI/OPEN	ENDED INFOR RETRI SYSTE INCLU SELEC DATA COLLE
0.466	AMDOA70210274BURCH/ROLE	FEDER GOVER INFOR SYSTE EDUCA

By comparing the two sets of search outputs corresponding to the two search requests that use different sets of term weights, one can easily draw the same conclusions as discussed in (I). Besides being very dependent on the term weights assigned by the user, the search process also depends to certain extent on the estimated recall and precision values supplied by the user. The major difference between the performance of the two algorithms may be summarized by stating that the more specific the search request, the more selective the search output will tend to be.

Note that the examples given above are one-parameter questions. In the case when more than one parameter is specified in one question, the parameters are treated as mutually exclusive; that is, each parameter is considered as one query vector. The final search output then consists of all the hits from the different query vectors. Further search examples are

included in Appendix E for reference.

5.3 Conclusions

It has been shown that both the optimum iterative feedback algorithm and the generalized optimum iterative feedback algorithm are capable of performing the retrieval of an optimum set of search output. One attraction of the algorithms is that no iterative search is necessary if the user's search request is already good enough. The only drawback is the additional search time required for iterative searches when a poor search request is encountered. However, in many cases, a maximum of two searches is probably sufficient. Therefore, the payoff of a poor search request in return for an optimum set of search output is after all not too discouraging.

It is worthwhile to note that the automatic indexing algorithm developed in this thesis may be further investigated for the possibility of an automatic generation of a thesaurus which plays an important role in modern information storage and retrieval. By converting the index term list and the significance values into a property vector, an algorithm can be developed for automatic recognition of synonyms. Lastly, the automatic indexing algorithm may prove to be very useful in the many applications of information handling systems.

BIBLIOGRAPHY

1. Cuadra, C. A. and Katter, R. V.: Opening the Black Box of Relevance, Journal of Documentation, 1967, Vol.23, No.4.

2. Lesk, M. E. and Salton, G.: Relevance Assessments and Retrieval System Evaluation, Information Storage and Retrieval, December 1968, Vol.4, No.2.

3. Salton, G.: Automatic Information Organization and Retrieval, McGraw Hill, New York, 1968.

4. Curtice, R. M. and Rosenberg, V.: Optimizing Retrieval Results with Man-Machine Interaction, Centre for the Information Sciences, Lehigh University, Bethlehem, Pa., 1965.

5. Rocchio, J. J. Jr.: Documentation Retrieval Systems - Optimization and Evaluation, Doctoral Thesis, Harvard University; Report ISR-10 to National Science Foundation, Harvard Computation Laboratory, March, 1966.

6. Salton, G.: Evaluation Problems in Interactive Information Retrieval, Information Storage and Retrieval, 1970, Vol.6, No.1, 29-44.

7. Heaps, H, S. and Ko, C. C.: Automatic Adaptive Processing of Questions in Documentation Retrieval, Proceedings of the American Society for Information Science, 1970, Vol.7, 319 321.

8. Ko, W. C. C.: Some Optimum Criteria for Information Retrieval, M.Sc. Thesis, University of Alberta, 1970.

9. Heaps, H. S.: Criteria for Optimum Effectiveness of Information Retrieval Systems, Information and Control, 1971, Vol.18, No.2, 156-167.

10. Heaps, H. S.: Boolean, Fractional, and Associative Searches on Truncated Title Words, Proceedings of the American Society for Information Science, Columbus,

Ohio, October 1968, Vol.5, 179-184.

11. Cagan, C.: A Highly Associative Document Retrieval System, Journal of the American Society for Information Science, Sept-Oct, 1970.

12. Jones, P. E. and Curtice, R. M.: A Framework for Comparing Term Association Measures, American Documentation, July, 1967.

13. Lesk, M. E.: Word-word Associations in Document Retrieval Systems, American Documentation, January, 1969.

14. Rosenberg, V.: A Study of Statistical Measures for Predicting Terms Used to Index Documents, Journal of the American Society for Information Science, Jan-Feb, 1971.

15. Salton, G. and Lesk, M. E.: Computer Evaluation of Indexing and Text Processing, Journal of the Association for Computing Machinery, Jan. 1968, Vol.5, No.1, 8-36.

16. Stiles, H. E.: The Association Factor in Information Retrieval, Journal of the ACM,, January 1961, Vol.8, No.2.

17. Rosenberg, V.: A Statistical Technique For Computer-Aided Indexing, Ph.D. Thesis, The University of Chicago, March, 1970.

18. Lo, A.: An Algorithm for Calculating the Product of a Large General Sparse Matrix and A Large Symmetric Sparse Matrix Using A Least Storage Scheme, University of Alberta Computing Review, 1972, 30-37.

19. Minsky, M. and Papert, S.: Perceptrons, The MIT Press, Massachusetts, 1969.

20. Nilsson, N.J.: Learning Machines, McGraw-Hill, New York, 1965.

APPENDIX A

COMPUTING SCIENCE DATA BASE OF JOURNAL ARTICLES
APPROXIMATELY 7,000 ARTICLES ON FILE, FEBRUARY 1972

AUTHOR NAMES AND TITLE WORDS TRUNCATED TO FIVE CHARACTERS
AUTHOR NAMES ARE FOLLOWED BY /
JOURNAL NAMES ARE REPRESENTED BY ASTM CODENS

CODEN	Journal Name (and period covered)
ACJ	Australian Computer Journal (1967-70). No ASTM coden.
AMDOA	American Documentation (1964-70).
ASLPA	Assoc. of Special Libraries and Informn. Bureau, Aslib Proc (1964-70)
ATCAA	Automatica (1964-69)
AURCA	Automation and Remote Control (1968-69).
BIT	Bit (1964-70). No ASTM coden.
CACMA	Communications of the ACM (1964-70).
CBMRB	Computers and Biomedical Research (1968-70).
CMPJA	Computer Journal (1964-70).
COBUA	Computer Bulletin (1965-70).
COMTB	Computing (1967-69).
DATMN	Datamation (1970). Non-ASTM coden used in error.
DTMNA	Datamation (1967-69).
ECECA	Economics Comp. and Econ. Cybernatics Studies and Research (1968-70).
ENCYA	Engineering Cybernatics (1969-70).
IBMJA	IBM Journal of Research and Development (1969-70).
IBMSA	International Business Machines, Systems Journal (1962-63).
ICCBA	ICC Bulletin (1964-67).
IETIA	IEEE Transactions on Information Theory (1969-70).
IFCNA	Information and Control (1964-70).
IFSRA	Information Storage and Retrieval (1966-69).
IJCMA	Informational Journal of Computer Mathematics (1968).
IJCOA	International Journal of Control (1969-70).
INPJB	Information Processing in Japan (1966-69).
ITCOB	IEEE Transactions on Computers (1969-70).
JACOA	Journal of the Association for Computing Machinery (1964 -70).
JCHDA	Journal of Chemical Documentation (1961-69).
JCSSB	Journal of Computer and Systems Sciences (1969).
JDOCA	Journal of Documentation (1963,1965-68).
JIMBA	SIAM Journal, Series B, Numerical Analysis (1969-70).
JLAUA	Journal of Library Automation (1968-70).
PAT	Pattern Recognition (1968-70). No ASTM coden.
PRITA	Problems of Information Transmission (1965-67).

APPENDIX B

The following statistical association measures are commonly used. The individual source is included in square brackets to the right. The interpretation of symbols is as follows:

c_{ij} = the extent to which $term_i$ is associated with $term_j$.

f_{ij} = the frequency of co-occurrence of $term_i$ and $term_j$.

f_i = the frequency of occurrence of $term_i$.

n = the total number of documents in the collection.

1. $c_{ij} = f_{ij}/(f_i+f_j-f_{ij})$. [12]

2. $c_{ij} = \log\{n(|nf_{ij}-f_if_j| -n/2)^2/f_if_j(n-f_i)(n-f_j)\}$. [16]

3. $c_{ij} = f_{ij}/f_i^{1/2}f_j^{1/2}$. [12]

4. $c_{ij} = f_{ij}/\min(f_i,f_j)-k[1/2-\min(f_i,f_j)/(f_i+f_j)]$. [11]

5. $c_{ij} = [f_{ij}-f_if_j/n]^2/[f_i-f_i^2/n][f_j-f_j^2/n]$. [14]

6. $c_{ij} = f_{ij}-f_if_j/n$. [12]

7. $c_{ij} = (f_{ij}-f_if_j/n)/(f_if_j/n)^{1/2}$. [12]

8. $c_{ij} = f_{ij}/f_j^k,\ 0\le k\le 1$. [12]

9. $c_{ij} = (f_{ij}-f_if_j/n)/[1-f_{ij}/(f_i+f_j)][f_i+f_j-f_if_j/n]$. [14]

10. $c_{ij} = \alpha f_{ij}+f_j,\ \alpha>\max f_j$. [14]

APPENDIX C

INDEX TERM LISTS

1. Index Term List Number One:

by using $c_{ij} = f_{ij}/(f_i+f_j-f_{ij})$.

SYSTE	1.000	COMPU	0.994	PROCE	0.980	TECHN	0.941
CONTR	0.939	GENER	0.931	TRANS	0.931	METHO	0.929
PROBL	0.921	INFOR	0.904	DATA	0.904	INTER	0.899
AUTOM	0.899	ANALY	0.897	OPTIM	0.891	FUNCT	0.890
LINEA	0.889	LANGU	0.887	STRUC	0.872	PROGR	0.866
ALGOR	0.861	TIME	0.859	CHEMI	0.858	INDEX	0.856
MULTI	0.853	RETRI	0.848	APPLI	0.846	ERROR	0.841
THEOR	0.839	MACHI	0.837	OPERA	0.833	CLASS	0.832
CODES	0.821	SOLUT	0.816	APPRO	0.809	CORRE	0.802
SEQUE	0.801	EVALU	0.801	EQUAT	0.795	DIGIT	0.795
VARIA	0.793	CONST	0.789	MATRI	0.787	LITER	0.786
INTEG	0.786	CONVE	0.784	SIMUL	0.784	DIFFE	0.781
EXPER	0.777	DESIG	0.775	STABI	0.768	STATE	0.756
NONLI	0.754	RANDO	0.749	AMERI	0.749	SIGNA	0.736
RECOG	0.735	REAL	0.728	PROBA	0.727	NUMER	0.723
LIBRA	0.722	LINE	0.722	NUMBE	0.720	FINIT	0.719
STAND	0.715	SCIEN	0.715	ESTIM	0.713	STATI	0.707
COMPA	0.706	COMMU	0.704	LOGIC	0.703	DYNAM	0.703
MODEL	0.703	GRAPH	0.702	PROPO	0.700	FORTR	0.697
ALGOL	0.695	ELEME	0.694	ORDER	0.693	ORGAN	0.693
CHANN	0.690	SEARC	0.689	PROPE	0.686	DETER	0.686
ITERA	0.686	CALCU	0.686	RELAT	0.685	MEMOR	0.684
UNIVE	0.681	SAMPL	0.680	DISTR	0.680	NOISE	0.679
FORMA	0.679	MICRO	0.678	FILTE	0.677	BOOLE	0.675
CIRCU	0.674	EFFIC	0.674	REGUL	0.673	MINIM	0.673
STORA	0.671	COORD	0.669	STOCH	0.668	FREE	0.665
BINAR	0.664	NOTAT	0.664	ALGEB	0.663	INPUT	0.662
SERVI	0.661	DISCR	0.659	NETWO	0.658	CODE	0.658
FEEDB	0.655	RESEA	0.654	LARGE	0.648	SCHEM	0.647
PARAM	0.644	CONDI	0.639	ABSTR	0.637	DEVEL	0.637
DEFIN	0.636	SIMPL	0.636	MAGNE	0.636	COMPL	0.635
SYNTA	0.634	SHARI	0.633	MEDIC	0.632	POLYN	0.632
SUBJE	0.632	CERTA	0.632	DIMEN	0.631	FORMU	0.629
CONTE	0.628	DIREC	0.627	DOCUM	0.626	SYNTH	0.626
ELECT	0.623	CURRE	0.622	POINT	0.620	BOUND	0.620
EFFEC	0.619	ORDIN	0.618	DISPL	0.617	COMPI	0.617
SELEC	0.616	MANAG	0.615	CHARA	0.614	SINGL	0.614
PRODU	0.613	INVES	0.611	IBM	0.611	TABLE	0.609
LIMIT	0.608	SPECI	0.606	ARITH	0.605	INVER	0.605
DECIS	0.603	BIT	0.602	CONTI	0.601	TAPE	0.597
REDUC	0.596	SOURC	0.595	FORM	0.594	VALUE	0.593
DERIV	0.592	STUDY	0.591	PERFO	0.591	ADAPT	0.590
SERIA	0.589	FREQU	0.587	PLATE	0.585	MODUL	0.585
ORIEN	0.583	MECHA	0.583	OUTPU	0.583	PATTE	0.583

TESTI	0.583	EIGEN	0.582	PRINT	0.582	HYBRI	0.582
BUSIN	0.582	PHASE	0.581	ENGIN	0.577	PRACT	0.577
SWITC	0.576	ECONO	0.572	BASED	0.572	FILE	0.571
SOCIA	0.570	ACTIV	0.568	PRESE	0.567	SQUAR	0.567
DESCR	0.566	ASYMP	0.564	SPECT	0.564	ACCES	0.563
NORMA	0.563	ABSOL	0.562	COLLE	0.561	PATEN	0.561
REPRE	0.560	MATHE	0.560	COMPO	0.560	TEST	0.560
FACTO	0.558	SERIE	0.556	THRES	0.554	CENTR	0.552
CATAL	0.551	SMALL	0.551	BLOCK	0.549	ASPEC	0.548
NON	0.546	SPACE	0.546	CONCE	0.546	IMPRO	0.545
USER	0.545	TYPE	0.544	ANALO	0.543	AMPLI	0.542
IDENT	0.542	EQUIV	0.542	SOFTW	0.542	CRITE	0.542
MANIP	0.541	INDUS	0.540	REPOR	0.538	GRAMM	0.538
SYMPO	0.538	SYNCH	0.538	CYCLI	0.538	NOTE	0.537
CENTE	0.531	PAPER	0.530	GROUP	0.530	CONFE	0.528
SYMME	0.528	RAPID	0.528	RESUL	0.527	SEPAR	0.526
SURVE	0.525	STUDI	0.525	RUNGE	0.524	NATIO	0.523
PHYSI	0.523	FIELD	0.522	DEVIC	0.521	ALPHA	0.520
BASE	0.519	QUADR	0.518	DEPAR	0.518	FILM	0.518
HARMO	0.518	PARSE	0.518	RECTA	0.518	BRIEF	0.515
PREFI	0.515	REDUN	0.515	FACT	0.515	ARGUM	0.515
COLLA	0.515	FREDH	0.515	INSUR	0.515	DEVIA	0.515
PL1	0.515	KONVE	0.515	CARD	0.514	LAW	0.514
CARDS	0.514	PARAL	0.514	COMME	0.514	VIEW	0.512
REFER	0.512	TERMI	0.512	DETEC	0.510	PUNCH	0.509
MONIT	0.506	CITAT	0.505	ADMIN	0.505	TEACH	0.504
EDUCA	0.501	SET	0.501	PURPO	0.500	AWARE	0.497
REMOT	0.496	CNLIN	0.496	QUANT	0.495	MARKE	0.494
PLANN	0.489	PERIO	0.488	PACKA	0.487	COMPR	0.485
LOCAL	0.485	RESPO	0.482	EXTRA	0.482	SECON	0.480
ASSIG	0.480	QUEUE	0.480	CODIN	0.478	HAND	0.478
PUBLI	0.478	PIECE	0.478	ASSOC	0.477	SETS	0.477
SENSI	0.476	REGIS	0.474	CONSI	0.473	RECUR	0.472
SOLVI	0.470	ROOTS	0.469	HEURI	0.469	GAMES	0.469
NATUR	0.468	LOOP	0.466	SPEED	0.466	GAUSS	0.465
PULSE	0.464	ASLIB	0.464	HANDL	0.463	PICTU	0.461
PRINC	0.460	PERMU	0.458	COEFF	0.456	FOURI	0.456
HIGH	0.456	CAPAC	0.454	DIVIS	0.454	QUASI	0.454
SELF	0.453	DIAGN	0.453	SUCCE	0.451	ALTER	0.451
FACIL	0.451	IMPLE	0.451	TERM	0.451	PROFE	0.450
ALLOC	0.450	RELEV	0.448	INVAR	0.447	STEP	0.444
CASE	0.444	FAST	0.443	POSIT	0.442	UTILI	0.442
INDEP	0.442	LINKS	0.442	DECOD	0.439	INTRO	0.439
QUALI	0.438	MARKO	0.436	PRECI	0.436	STRAT	0.434
FLOWC	0.434	CHART	0.433	TOWAR	0.433	RELAY	0.432
ADDIT	0.432	FUNDA	0.432	REQUI	0.431	EXPAN	0.431
REALI	0.430	ASA	0.430	BOCKS	0.428	EQUIP	0.428
WRITI	0.428	PREDI	0.428	DUAL	0.426	JOURN	0.426
360	0.422	NETS	0.422	DEPEN	0.418	SHIFT	0.418
RESOU	0.417	DECOM	0.417	PREPA	0.415	CHEBY	0.414
REGIO	0.413	CONDU	0.412	WEIGH	0.411	TITLE	0.411
PSEUD	0.411	REVIE	0.410	VECTO	0.408	ACCOU	0.408
DELAY	0.407	RIGHT	0.405	RANK	0.405	PASS	0.405
DOMAI	0.405	FINDI	0.405	CURRI	0.404	LIST	0.404
COBOL	0.403	BIBLI	0.401	ARTIC	0.401	SCHOO	0.399

EXTEN	0.399	MOTIO	0.398	COST	0.398	PERSO	0.398
DIAGR	0.398	LEVEL	0.398	PLANE	0.396	INQUI	0.396
DISCO	0.395	MAJOR	0.395	SORT	0.395	SUM	0.395
WATER	0.395	JAPAN	0.395	ACADE	0.395	SERVO	0.395
NONPA	0.395	KDF9	0.395	KEYWO	0.395	TUTOR	0.395
ANNOU	0.395	EXECU	0.395	BOARD	0.395	DISTO	0.395
PAGED	0.395	FIXED	0.395	GROWT	0.395	SCHED	0.395
EXTRE	0.394	PHRAS	0.394	SYMBO	0.393	SUBRE	0.392
SCALE	0.392	STIBI	0.388	REMAR	0.387	MEETI	0.387
ORTHO	0.386	CHOIC	0.385	ELIMI	0.385	PRIOR	0.384
LENGT	0.383	SUBMI	0.378	BELL	0.378	BACKW	0.378
PROBS	0.378	COURT	0.378	BORDE	0.378	POLLU	0.378
CEP	0.378	RECEP	0.378	PROMO	0.378	MONOI	0.378
PUFFT	0.378	CALL	0.378	JORDA	0.378	DREDG	0.378
RIGID	0.378	POLES	0.378	ALLIE	0.378	CULTU	0.378
CHOMS	0.378	POLIT	0.378	ERRAT	0.378	BIVAR	0.378
UNSTE	0.378	PREFE	0.378	BOND	0.378	POSTA	0.378
STEPP	0.378	LOCUS	0.378	AERON	0.378	TANK	0.378
BIRTH	0.378	RCA	0.378	SHOCK	0.378	DECOC	0.378
SIMON	0.378	PROLE	0.378	JACKS	0.378	CDS	0.378
APPRA	0.378	RAY	0.378	TRI	0.378	RACHF	0.378
REPLA	0.378	MARK	0.378	HOBBS	0.378	DEMON	0.378
HOT	0.378	RC400	0.378	STEPS	0.378	DISCI	0.378
GAIN	0.378	PREVI	0.378	HELIC	0.378	DOMIN	0.378
FIRM	0.378	PRICI	0.378	SESSI	0.378	REGEL	0.378
TROUB	0.378	DORN	0.378	POSTI	0.378	SLT	0.378
LATEN	0.378	PEACE	0.378	RENAM	0.378	ECMA	0.378
UNSUP	0.378	PAGE	0.378	PLAN	0.378	CANAD	0.378
ORANI	0.378	ONTAR	0.378	SYNAP	0.378	PEEKA	0.378
TRIAN	0.378	ELLIO	0.378	MAGNU	0.378	SIZED	0.378
SUBOP	0.378	PARIT	0.378	TELEC	0.378	ENDED	0.378
SYLLA	0.378	PROCR	0.378	NONCO	0.378	BANDS	0.378
SIZE	0.378	TRACK	0.378	PARTY	0.378	CORNE	0.378
MARKU	0.378	ACOUT	0.378	STAR	0.378	SHEFF	0.378
OCCUR	0.378	ACMCP	0.378	EXCLU	0.378	MEANI	0.378
ORBIT	0.378	AGENT	0.378	BITS	0.378	ROMAN	0.378
PACKE	0.378	PANTA	0.378	OHIO	0.378	PROGA	0.378
MERGI	0.378	ANTIC	0.378	CANCE	0.378	ORDNU	0.378
SCHOL	0.378	FIT	0.378	TRIAL	0.378	I/O	0.378
AMEND	0.378	FCC	0.378	VIROL	0.378	OPTIO	0.378
MERCU	0.378	PANEL	0.378	ATTEM	0.378	DIGES	0.378
CHICA	0.378	SHOP	0.378	TREAT	0.378	LOCI	0.378
COLCU	0.378	CONFR	0.378	LOAD	0.378	CARTO	0.378
APPRE	0.378	UNSTA	0.378	LONDO	0.378	LOSSE	0.378
BROMB	0.378	LANCZ	0.378	APL	0.378	SHELL	0.378
SPATI	0.378	MARKS	0.378	BASSA	0.378	JUMP	0.378
BINDI	0.378	MASTE	0.378	VOYSE	0.378	BAIRS	0.378
VISIO	0.378	MANUS	0.378	WILSO	0.378	ROSEN	0.378
DUPLE	0.378	SIDES	0.378	2314	0.378	KOSHE	0.378
INCOR	0.378	GIER	0.378	LABS	0.378	UNCON	0.378
ATTEN	0.378	LANGE	0.378	UNVOI	0.378	BROWN	0.378
DISAG	0.378	AUTON	0.378	AUDIO	0.378	LAYOU	0.378
TOWER	0.378	HOUSI	0.378	ANCMA	0.378	FIDEL	0.378
RESIS	0.378	HILL	0.378	ROBIN	0.378	KNUTH	0.378
SARDI	0.378	LENSE	0.378	DUALI	0.378	SEMIG	0.378

ROYAL	0.378	STOPP	0.378	SIDE	0.378	GROSS	0.378
FERRI	0.378	NORMS	0.378	SUBSE	0.378	BANKI	0.378
TANDE	0.378	NETHE	0.378	SLCW	0.378	NET	0.378
STAT1	0.378	META	0.378	CLENS	0.378	NONSI	0.378
HAMMI	0.378	BALL	0.378	NONDE	0.378	NONRE	0.378
ICL	0.378	REWRI	0.378	MILNE	0.378	FLUX	0.378
MODAL	0.378	BEREC	0.378	MORPH	0.378	MODES	0.378
IMPRE	0.378	BLIND	0.378	METAB	0.378	MATEM	0.378
DECAD	0.378	GE	0.378	MERGE	0.378	MERSE	0.378
IONIN	0.378	INTEN	0.378	INTEL	0.378	PREVE	0.378
JENKI	0.378	EXPOR	0.378	IRRED	0.378	IONES	0.378
INDIR	0.378	GIRO	0.378	KEIO	0.378	POLYP	0.378
LAGS	0.378	ENZYM	0.378	ISSUE	0.378	JONES	0.378
HARD	0.378	ELEKT	0.378	INACC	0.378	RAPHS	0.378
HIRSC	0.378	DAVIE	0.378	ILL	0.378	ICT	0.378
HADAM	0.378	RETAI	0.378	HIGHE	0.378	REED	0.378
REGIM	0.378	HOLLA	0.378	HOHER	0.378	EASTM	0.378
LONGE	0.378	GIVEN	0.378	PAREN	0.378	ESSO	0.378
FREED	0.378	FIXAT	0.378	RISK	0.378	HASH	0.378
PERT	0.378	METNO	0.378	HIDDE	0.378	HABIT	0.378
HETER	0.378	DIODE	0.378	PENTO	0.378	FRANK	0.378
DOUGL	0.378	ISOMC	0.378	FLOWS	0.378	FRANC	0.378
DEEP	0.378	IMAGE	0.378	FLOWR	0.378	GAAS	0.378
MINER	0.378	OVERD	0.378	GLYCO	0.378	FLORE	0.378
DRUGS	0.378	BIGEL	0.378	EINSC	0.378	LOADI	0.378
EMPHA	0.378	ESTAB	0.378	EDELM	0.378	EXCHA	0.378
BETA	0.378	FABRI	0.378	MAPS	0.378	FLEXI	0.378
FEASI	0.378	ESSAY	0.378	DEFLE	0.378	AREA	0.378
DECID	0.378	KNIGH	0.378	DISSI	0.378	PRACN	0.378
DRIFT	0.378	PURSU	0.378	DATAN	0.378	LAYMA	0.378
DATAF	0.378	LIBER	0.378	DEFER	0.378	FIGUR	0.378
CLARI	0.378	MUSIC	0.378	MOORE	0.378	DEPOS	0.378
SIGNE	0.378	NICHO	0.378	NORM	0.378	DONNE	0.378
CBAC	0.378	MOD	0.378	MAEHL	0.378	DATAM	0.378
LAGRA	0.378	CREDI	0.378	MICHI	0.378	COSMI	0.378
SINGU	0.377	MODIF	0.377	FLCW	0.375	GUIDA	0.374
WORKI	0.374	IMPUL	0.370	CAVIT	0.367	RESER	0.367
RACHE	0.367	STORE	0.367	MOVIN	0.367	ASYMM	0.367
KORZH	0.367	GRAHA	0.367	IMPED	0.367	DEMOD	0.367
EVIDE	0.367	TAU	0.367	BRAIN	0.367	CONFO	0.367
CHURC	0.367	BENDI	0.367	BLEND	0.367	AUTO	0.367
AREAS	0.367	UNFOR	0.367	TOLER	0.367	TELEM	0.367
SIMSC	0.367	WESCO	0.367	VERIE	0.367	VIEWP	0.367
STAGE	0.367	SUBCO	0.367	RETRC	0.367	SORTE	0.367
SCALI	0.367	ROTAT	0.367	DISSE	0.367	THESA	0.367
TREE	0.367	FURTH	0.361	ENCOD	0.361	LAPLA	0.360
POSSI	0.356	ATLAS	0.356	TIMET	0.355	EXPRE	0.355
KUTTA	0.355	SENSE	0.351	EVENT	0.345	NUCLE	0.345
REGAR	0.343	POLE	0.343	PEOPL	0.343	DECEN	0.343
FILLI	0.343	UNIFO	0.342	INSTR	0.339		

TOTAL = 815

2. Index Term List Number Two:

by using $c_{ij} = f_{ij}/f_i^{1/2} f_j^{1/2}$.

SYSTE	1.000	COMPU	0.988	PROCE	0.950	CONTR	0.908
METHO	0.906	TECHN	0.903	GENER	0.897	TRANS	0.894
PROBL	0.888	INFOR	0.880	AUTOM	0.870	DATA	0.868
ANALY	0.864	INTER	0.857	OPTIM	0.856	FUNCT	0.854
LINEA	0.852	LANGU	0.846	PROGR	0.838	STRUC	0.835
TIME	0.826	CHEMI	0.824	ALGOR	0.823	INDEX	0.815
MULTI	0.814	RETRI	0.808	APPLI	0.806	THEOR	0.805
ERROR	0.802	MACHI	0.798	OPERA	0.795	CLASS	0.790
CODES	0.786	SOLUT	0.785	APPRO	0.772	CORRE	0.769
EVALU	0.766	EQUAT	0.765	SEQUE	0.765	DIGIT	0.756
MATRI	0.756	VARIA	0.754	SIMUL	0.751	LITER	0.750
CONST	0.749	DIFFE	0.748	INTEG	0.748	CONVE	0.748
EXPER	0.742	DESIG	0.733	STABI	0.732	NONLI	0.725
AMERI	0.724	STATE	0.721	RANDO	0.719	SIGNA	0.710
RECOG	0.703	PROBA	0.698	LINE	0.696	REAL	0.695
FINIT	0.693	LIBRA	0.691	NUMBE	0.691	STAND	0.689
SCIEN	0.687	NUMER	0.687	STATI	0.681	COMMU	0.680
LOGIC	0.679	PROPO	0.679	DYNAM	0.678	ESTIM	0.678
FORTR	0.677	COMPA	0.677	ELEME	0.672	ALGOL	0.670
GRAPH	0.669	ORDER	0.667	SEARC	0.665	MODEL	0.664
ITERA	0.663	ORGAN	0.663	DETER	0.662	MEMOR	0.662
CHANN	0.661	CALCU	0.661	PROPE	0.660	RELAT	0.658
NOISE	0.658	FILTE	0.657	SAMPL	0.657	DISTR	0.655
UNIVE	0.652	FORMA	0.652	REGUL	0.652	BOOLE	0.652
MICRO	0.651	MINIM	0.650	STORA	0.650	CIRCU	0.649
EFFIC	0.647	FREE	0.644	COORD	0.644	NOTAT	0.643
STOCH	0.643	SERVI	0.639	ALGEB	0.639	INPUT	0.638
DISCR	0.636	BINAR	0.631	RESEA	0.630	NETWO	0.627
SCHEM	0.626	CODE	0.625	FEEDB	0.625	LARGE	0.624
CONDI	0.621	PARAM	0.619	DEFIN	0.618	MAGNE	0.616
DEVEL	0.615	CERTA	0.615	COMPL	0.614	DIMEN	0.614
SYNTA	0.613	POLYN	0.613	SIMPL	0.613	SUBJE	0.611
MEDIC	0.611	ABSTR	0.611	CONTE	0.609	SHARI	0.605
DOCUM	0.604	ELECT	0.603	EFFEC	0.602	POINT	0.602
DIREC	0.601	FORMU	0.601	CURRE	0.600	ORDIN	0.599
BOUND	0.596	DISPL	0.596	SELEC	0.595	PRODU	0.594
COMPI	0.594	SYNTH	0.593	SINGL	0.593	INVES	0.593
BIT	0.591	TABLE	0.591	CHARA	0.590	MANAG	0.589
LIMIT	0.587	IBM	0.586	INVER	0.586	ARITH	0.583
SPECI	0.581	REDUC	0.581	CONTI	0.580	FORM	0.579
PERFO	0.577	VALUE	0.576	TAPE	0.576	STUDY	0.576
SOURC	0.575	DERIV	0.573	DECIS	0.573	BUSIN	0.571
PRINT	0.571	HYBRI	0.571	ADAPT	0.569	SERIA	0.569
FREQU	0.568	MODUL	0.567	OUTPU	0.565	ORIEN	0.563
MECHA	0.562	EIGEN	0.562	PATTE	0.562	PLATE	0.562
TESTI	0.560	PRACT	0.560	ENGIN	0.559	PHASE	0.556
SQUAR	0.554	FILE	0.554	SWITC	0.553	SPECT	0.552
ACTIV	0.551	SOCIA	0.549	TEST	0.549	ECONO	0.548

BASED	0.547	DESCR	0.546	NORMA	0.546	PRESE	0.546
PATEN	0.545	COMPO	0.544	ASYMP	0.543	ABSOL	0.542
FACTO	0.542	COLLE	0.542	SMALL	0.541	REPRE	0.541
ACCES	0.541	THRES	0.537	SERIE	0.536	MATHE	0.536
IMPRO	0.535	CENTR	0.533	NON	0.532	SPACE	0.530
TYPE	0.530	USER	0.530	AMPLI	0.529	CATAL	0.529
EQUIV	0.529	IDENT	0.529	ASPEC	0.529	CONCE	0.528
CYCLI	0.527	BLOCK	0.527	ANALO	0.527	SYMPO	0.524
MANIP	0.524	SOFTW	0.521	CRITE	0.521	INDUS	0.521
PAPER	0.519	REPOR	0.519	NOTE	0.519	SYNCH	0.518
SEPAR	0.516	GRAMM	0.516	RAPID	0.511	CENTE	0.511
CONFE	0.510	STUDI	0.509	PHYSI	0.509	GROUP	0.508
SYMME	0.508	RESUL	0.507	RUNGE	0.506	DEVIC	0.506
SURVE	0.505	ALPHA	0.503	BASE	0.502	NATIO	0.500
COMME	0.499	FIELD	0.499	QUADR	0.498	LAW	0.498
CARD	0.498	CARDS	0.498	ARGUM	0.497	PREFI	0.497
FACT	0.497	FREDH	0.497	INSUR	0.497	DEVIA	0.497
REDUN	0.497	PL1	0.497	BRIEF	0.497	KONVE	0.497
COLLA	0.497	DEPAR	0.497	FILM	0.497	HARMO	0.497
PARSE	0.497	RECTA	0.497	REFER	0.495	TERMI	0.495
VIEW	0.495	PARAL	0.493	MONIT	0.491	DETEC	0.490
TEACH	0.489	PUNCH	0.489	EDUCA	0.488	ADMIN	0.487
CITAT	0.487	PURPO	0.484	SET	0.482	QUANT	0.482
AWARE	0.481	REMOT	0.479	MARKE	0.477	PERIO	0.475
ONLIN	0.474	PLANN	0.473	COMPR	0.472	LOCAL	0.472
PACKA	0.471	PIECE	0.467	REGIS	0.465	EXTRA	0.465
RESPO	0.465	ASSIG	0.464	QUEUE	0.464	ASSOC	0.464
SENSI	0.463	SECON	0.462	HAND	0.459	SETS	0.459
PUBLI	0.458	CODIN	0.458	LOOP	0.458	GAMES	0.457
ROOTS	0.456	HEURI	0.456	RECUR	0.455	CONSI	0.455
HANDL	0.454	SOLVI	0.453	PRINC	0.453	SPEED	0.452
NATUR	0.450	ASLIB	0.449	GAUSS	0.448	PULSE	0.447
SELF	0.445	CAPAC	0.445	FACIL	0.444	PICTU	0.443
PERMU	0.441	HIGH	0.440	DIAGN	0.439	TERM	0.438
DIVIS	0.438	QUASI	0.438	FOURI	0.438	COEFF	0.438
IMPLE	0.434	ALTER	0.434	SUCCE	0.434	ALLOC	0.433
RELEV	0.433	PROFE	0.432	INVAR	0.430	STEP	0.429
FAST	0.428	CASE	0.426	POSIT	0.425	INTRO	0.425
INDEP	0.425	UTILI	0.424	LINKS	0.424	QUALI	0.423
MARKO	0.422	STRAT	0.422	PRECI	0.421	DECOD	0.421
PREDI	0.418	FLOWC	0.417	RELAY	0.417	ADDIT	0.417
FUNDA	0.417	TOWAR	0.417	CHART	0.417	ASA	0.416
REQUI	0.416	EXPAN	0.416	REALI	0.412	BOOKS	0.412
EQUIP	0.412	WRITI	0.412	JOURN	0.411	DUAL	0.408
NETS	0.407	RESOU	0.406	360	0.406	DEPEN	0.405
SHIFT	0.405	DECOM	0.403	CHEBY	0.400	REGIO	0.399
PREPA	0.399	WEIGH	0.399	CONDU	0.396	VECTO	0.396
TITLE	0.396	DELAY	0.393	REVIE	0.392	PSEUD	0.392
ACCOU	0.390	RANK	0.389	RIGHT	0.389	PASS	0.389
DOMAI	0.389	CURRI	0.389	FINDI	0.389	LIST	0.388
ARTIC	0.388	LEVEL	0.386	DIAGR	0.386	BIBLI	0.386
COBOL	0.386	EXTEN	0.385	SCHOO	0.385	MOTIO	0.381
COST	0.381	PERSO	0.381	PLANE	0.381	INQUI	0.381
SCHED	0.380	SCALE	0.380	MAJOR	0.380	DISCO	0.380
SORT	0.380	SUM	0.380	WATER	0.380	JAPAN	0.380

ACADE	0.380	SERVO	0.380	NONPA	0.380	KDF9	0.380
KEYWO	0.380	TUTOR	0.380	ANNOU	0.380	EXECU	0.380
BOARD	0.380	DISTO	0.380	PAGED	0.380	FIXED	0.380
GROWT	0.380	PHRAS	0.380	EXTRE	0.380	SUBRE	0.378
SYMBO	0.376	STIBI	0.371	REMAR	0.371	MEETI	0.371
CHOIC	0.370	ELIMI	0.370	PRIOR	0.370	ORTHO	0.369
LENGT	0.368	MODIF	0.365	SUBMI	0.361	BELL	0.361
BACKW	0.361	PROBS	0.361	COURT	0.361	BORDE	0.361
POLLU	0.361	CEP	0.361	RECEP	0.361	PROMO	0.361
MONOI	0.361	PUFFT	0.361	CALL	0.361	JORDA	0.361
DREDG	0.361	RIGID	0.361	POLES	0.361	ALLIE	0.361
CULTU	0.361	CHOMS	0.361	POLIT	0.361	ERRAT	0.361
BIVAR	0.361	UNSTE	0.361	PREFE	0.361	BOND	0.361
POSTA	0.361	STEPP	0.361	LOCUS	0.361	AERON	0.361
TANK	0.361	BIRTH	0.361	RCA	0.361	SHOCK	0.361
DECOC	0.361	SIMON	0.361	PROLE	0.361	JACKS	0.361
CDS	0.361	APPRA	0.361	RAY	0.361	TRI	0.361
RACHF	0.361	REPLA	0.361	MARK	0.361	HOBBS	0.361
DEMON	0.361	HOT	0.361	RC400	0.361	STEPS	0.361
DISCI	0.361	GAIN	0.361	PREVI	0.361	HELIC	0.361
DOMIN	0.361	FIRM	0.361	PRICI	0.361	SESSI	0.361
REGEL	0.361	TROUB	0.361	DORN	0.361	POSTI	0.361
SLT	0.361	LATEN	0.361	PEACE	0.361	RENAM	0.361
ECMA	0.361	UNSUP	0.361	PAGE	0.361	PLAN	0.361
CANAD	0.361	ORANI	0.361	ONTAR	0.361	SYNAP	0.361
PEEKA	0.361	TRIAN	0.361	ELLIC	0.361	MAGNU	0.361
SIZED	0.361	SUBOP	0.361	PARIT	0.361	TELEC	0.361
ENDED	0.361	SYLLA	0.361	PROCR	0.361	NONCO	0.361
BANDS	0.361	SIZE	0.361	TRACK	0.361	PARTY	0.361
CORNE	0.361	MARKU	0.361	ACOUT	0.361	STAR	0.361
SHEFF	0.361	OCCUR	0.361	ACMCP	0.361	EXCLU	0.361
MEANI	0.361	ORBIT	0.361	AGENT	0.361	BITS	0.361
ROMAN	0.361	PACKE	0.361	PANTA	0.361	OHIO	0.361
PROGA	0.361	MERGI	0.361	ANTIC	0.361	CANCE	0.361
ORDNU	0.361	SCHOL	0.361	FIT	0.361	TRIAL	0.361
I/O	0.361	AMEND	0.361	FCC	0.361	VIROL	0.361
OPTIO	0.361	MERCU	0.361	PANEL	0.361	ATTEM	0.361
DIGES	0.361	CHICA	0.361	SHOP	0.361	TREAT	0.361
LOCI	0.361	COLOU	0.361	CONFR	0.361	LOAD	0.361
CARTO	0.361	APPRE	0.361	UNSTA	0.361	LONDO	0.361
LOSSE	0.361	BROMB	0.361	LANCZ	0.361	APL	0.361
SHELL	0.361	SPATI	0.361	MARKS	0.361	BASSA	0.361
JUMP	0.361	BINDI	0.361	MASTE	0.361	VOYSE	0.361
BAIRS	0.361	VISIO	0.361	MANUS	0.361	WILSO	0.361
ROSEN	0.361	DUPLE	0.361	SIDES	0.361	2314	0.361
KOSHE	0.361	INCOR	0.361	GIER	0.361	LABS	0.361
UNCON	0.361	ATTEN	0.361	LANGE	0.361	UNVOI	0.361
BROWN	0.361	DISAG	0.361	AUTON	0.361	AUDIO	0.361
LAYOU	0.361	TOWER	0.361	HOUSI	0.361	ANOMA	0.361
FIDEL	0.361	RESIS	0.361	HILL	0.361	ROBIN	0.361
KNUTH	0.361	SARDI	0.361	LENSE	0.361	DUALI	0.361
SEMIG	0.361	ROYAL	0.361	STOPP	0.361	SIDE	0.361
GROSS	0.361	FERRI	0.361	NORMS	0.361	SUBSE	0.361
BANKI	0.361	TANDE	0.361	NETHE	0.361	SLOW	0.361
NET	0.361	STAT1	0.361	META	0.361	CLENS	0.361

NONSI	0.361	HAMMI	0.361	BALL	0.361	NONDE	0.361
NONRF	0.361	ICL	0.361	REWRI	0.361	MILNE	0.361
FLUX	0.361	MODAL	0.361	BEREC	0.361	MORPH	0.361
MODES	0.361	IMPRE	0.361	BLIND	0.361	METAB	0.361
MATEM	0.361	DECAD	0.361	GE	0.361	MERGE	0.361
MERSE	0.361	IONIN	0.361	INTEN	0.361	INTEL	0.361
PREVE	0.361	JENKI	0.361	EXPOR	0.361	IRRED	0.361
IONES	0.361	INDIR	0.361	GIRO	0.361	KEIO	0.361
POLYP	0.361	LAGS	0.361	ENZYM	0.361	ISSUE	0.361
JONES	0.361	HARD	0.361	ELEKT	0.361	INACC	0.361
RAPHS	0.361	HIRSC	0.361	DAVIE	0.361	ILL	0.361
ICT	0.361	HADAM	0.361	RETAI	0.361	HIGHE	0.361
REED	0.361	REGIM	0.361	HOLLA	0.361	HOHER	0.361
EASTM	0.361	LONGE	0.361	GIVEN	0.361	PAREN	0.361
ESSO	0.361	FREED	0.361	FIXAT	0.361	RISK	0.361
HASH	0.361	PERT	0.361	METNO	0.361	HIDDE	0.361
HABIT	0.361	HETER	0.361	DIODE	0.361	PENTO	0.361
FRANK	0.361	DOUGL	0.361	ISOMO	0.361	FLOWS	0.361
FRANC	0.361	DEEP	0.361	IMAGE	0.361	FLOWR	0.361
GAAS	0.361	MINER	0.361	OVERD	0.361	GLYCO	0.361
FLORE	0.361	DRUGS	0.361	BIGEL	0.361	EINSC	0.361
LOADI	0.361	EMPHA	0.361	ESTAB	0.361	EDELM	0.361
EXCHA	0.361	BETA	0.361	FABRI	0.361	MAPS	0.361
FLEXI	0.361	FEASI	0.361	ESSAY	0.361	DEFLE	0.361
AREA	0.361	DECID	0.361	KNIGH	0.361	DISSI	0.361
PRACN	0.361	DRIFT	0.361	PURSU	0.361	DATAN	0.361
LAYMA	0.361	DATAF	0.361	LIBER	0.361	DEFER	0.361
FIGUR	0.361	CLARI	0.361	MUSIC	0.361	MOORE	0.361
DEPOS	0.361	SIGNE	0.361	NICHO	0.361	NORM	0.361
DONNE	0.361	CBAC	0.361	MOD	0.361	MAEHL	0.361
DATAM	0.361	LAGRA	0.361	CREDI	0.361	MICHI	0.361
COSMI	0.361	FLOW	0.360	GUIDA	0.360	WORKI	0.360
SINGU	0.359	IMPUL	0.357	CAVIT	0.351	RESER	0.351
RACHE	0.351	STORE	0.351	MOVIN	0.351	ASYMM	0.351
KORZH	0.351	GRAHA	0.351	IMPED	0.351	DEMOD	0.351
EVIDE	0.351	TAU	0.351	BRAIN	0.351	CONFO	0.351
CHURC	0.351	BENDI	0.351	BLEND	0.351	AUTO	0.351
AREAS	0.351	UNFOR	0.351	TOLER	0.351	TELEM	0.351
SIMSC	0.351	WESCO	0.351	VERTE	0.351	VIEWP	0.351
STAGE	0.351	SUBCO	0.351	RETRO	0.351	SORTE	0.351
SCALI	0.351	ROTAT	0.351	DISSE	0.351	THESA	0.351
TREE	0.351	FURTH	0.348	ENCOD	0.348	LAPLA	0.347
POSSI	0.341	ATLAS	0.341	EXPRE	0.341	KUTTA	0.341
TIMET	0.341	SENSE	0.338	EVENT	0.331	REGAR	0.330
POLE	0.330	PEOPL	0.330	DECEN	0.330	FILLI	0.330
NUCLE	0.330	UNIFO	0.329	INSTR	0.327	ANSWE	0.324

TOTAL = 816

3. <u>Index Term List Number Three:</u>

by using $c_{ij} = [f_{ij} - f_i f_j / n]^2 / [f_i - f_i^2 / n][f_j - f_j^2 / n]$.

COMPU	1.000	SYSTE	0.976	PROCE	0.974	CONTR	0.946
TECHN	0.936	TRANS	0.928	METHO	0.905	DATA	0.904
OPTIM	0.899	GENER	0.897	INFOR	0.893	PROBL	0.891
LANGU	0.889	FUNCT	0.888	AUTOM	0.879	LINEA	0.873
INTER	0.867	ALGOR	0.849	CHEMI	0.849	STRUC	0.847
ANALY	0.843	PROGR	0.840	INDEX	0.838	MULTI	0.834
CODES	0.821	OPERA	0.820	CLASS	0.811	RETRI	0.811
ERROR	0.805	APPRO	0.804	EVALU	0.802	TIME	0.802
DIGIT	0.802	THEOR	0.801	EQUAT	0.798	DIFFE	0.797
APPLI	0.794	SEQUE	0.791	SOLUT	0.788	INTEG	0.785
CONVE	0.784	LITER	0.783	VARIA	0.780	MATRI	0.780
AMERI	0.779	CONST	0.779	CORRE	0.779	EXPER	0.778
STABI	0.778	SIMUL	0.767	MACHI	0.767	RANDO	0.761
SIGNA	0.757	LINE	0.745	RECOG	0.745	NONLI	0.745
REAL	0.742	DESIG	0.736	FINIT	0.735	SCIEN	0.735
ALGOL	0.730	STATI	0.730	LIBRA	0.728	LOGIC	0.727
PROBA	0.726	STAND	0.724	STATE	0.722	NUMER	0.717
NUMBE	0.716	RELAT	0.715	FORTR	0.715	PROPO	0.715
SEARC	0.714	DETER	0.712	COMPA	0.712	MEMOR	0.712
ELEME	0.711	UNIVE	0.711	SAMPL	0.711	COMMU	0.710
GRAPH	0.710	REGUL	0.709	INPUT	0.709	MICRO	0.709
DYNAM	0.709	ESTIM	0.706	BOOLE	0.706	EFFIC	0.705
CALCU	0.700	ORDER	0.698	FORMA	0.697	MODEL	0.697
PROPE	0.695	COORD	0.694	DISCR	0.694	CONDI	0.693
FILTE	0.691	LARGE	0.691	DISTR	0.691	STORA	0.686
MINIM	0.686	NOISE	0.685	CERTA	0.684	SERVI	0.682
CHANN	0.681	NOTAT	0.680	STOCH	0.679	MAGNE	0.679
FREE	0.678	BINAR	0.677	SUBJE	0.675	MEDIC	0.675
CIRCU	0.675	DIREC	0.673	ELECT	0.672	ABSTR	0.666
POLYN	0.665	NETWO	0.663	ITERA	0.662	RESEA	0.661
ORGAN	0.661	DOCUM	0.660	FORMU	0.659	POINT	0.657
DIMEN	0.656	ALGEB	0.655	CURRE	0.654	COMPI	0.654
CODE	0.653	DEVEL	0.651	SPECI	0.649	SINGL	0.648
COMPL	0.648	INVER	0.648	ORDIN	0.647	DISPL	0.647
PARAM	0.646	DEFIN	0.646	FEEDB	0.645	REDUC	0.644
SCHEM	0.643	SERIA	0.641	BIT	0.641	TABLE	0.640
SIMPL	0.639	BOUND	0.639	INVES	0.639	ADAPT	0.639
LIMIT	0.639	SHARI	0.639	PLATE	0.638	IBM	0.637
DERIV	0.634	MECHA	0.631	TAPE	0.631	CONTI	0.630
PATTE	0.629	CONTE	0.627	ARITH	0.627	ORIEN	0.627
STUDY	0.627	CHARA	0.626	VALUE	0.625	SYNTA	0.625
MANAG	0.625	TESTI	0.624	SYNTH	0.624	FORM	0.621
SOCIA	0.620	PERFO	0.619	MODUL	0.619	HYBRI	0.618
PRINT	0.618	BUSIN	0.618	SWITC	0.618	EIGEN	0.616
OUTPU	0.615	ASYMP	0.614	PHASE	0.613	ECONO	0.613
ABSOL	0.611	COLLE	0.611	EFFEC	0.611	DECIS	0.609
BASED	0.609	FREQU	0.608	ACCES	0.607	PRODU	0.607
PRACT	0.607	FILE	0.604	CENTR	0.602	REPRE	0.600

ENGIN	0.597	SOURC	0.597	THRES	0.595	COMPO	0.595
BLOCK	0.594	SQUAR	0.591	DESCR	0.591	PRESE	0.590
CRITE	0.590	CATAL	0.590	ACTIV	0.588	MATHE	0.587
TEST	0.587	SERIE	0.586	SPECT	0.584	SOFTW	0.582
NORMA	0.582	FACTO	0.580	ASPEC	0.580	USER	0.578
GRAMM	0.576	MANIP	0.575	PATEN	0.573	SMALL	0.572
IMPRO	0.571	SYNCH	0.571	NOTE	0.571	RAPID	0.570
INDUS	0.569	SEPAR	0.569	SELEC	0.568	SYMME	0.567
RUNGE	0.567	SPACE	0.567	FIELD	0.565	AMPLI	0.565
IDENT	0.564	EQUIV	0.564	RECTA	0.564	FILM	0.564
DEPAR	0.564	HARMO	0.564	PARSE	0.564	CONFE	0.563
RESUL	0.563	INSUR	0.562	FREDH	0.562	REDUN	0.562
PREFI	0.562	BRIEF	0.562	DEVIA	0.562	FACT	0.562
PL1	0.562	ARGUM	0.562	KONVE	0.562	COLLA	0.562
ALPHA	0.561	CYCLI	0.560	GROUP	0.560	TYPE	0.559
BASE	0.559	SYMPO	0.558	TERMI	0.554	REFER	0.554
REPOR	0.553	PUNCH	0.553	NATIO	0.549	ADMIN	0.549
LAW	0.549	CARD	0.549	CARDS	0.549	NON	0.549
CITAT	0.548	VIEW	0.548	CENTE	0.544	DEVIC	0.543
SET	0.543	AWARE	0.543	PARAL	0.542	QUADR	0.541
STUDI	0.540	SURVE	0.539	TEACH	0.538	REMOT	0.537
COMME	0.537	CONCE	0.536	QUANT	0.536	PURPO	0.535
ONLIN	0.533	ANALO	0.532	DETEC	0.531	MARKE	0.529
PIECE	0.527	MONIT	0.526	PACKA	0.524	PLANN	0.523
HAND	0.522	QUEUE	0.521	ASSIG	0.521	COMPR	0.521
RESPO	0.520	EXTRA	0.520	SETS	0.519	CODIN	0.516
SENSI	0.513	LOCAL	0.513	ASSOC	0.513	CONSI	0.512
ASLIB	0.506	PUBLI	0.505	RECUR	0.505	NATUR	0.502
GAMES	0.502	PULSE	0.501	REGIS	0.501	PICTU	0.500
ROOTS	0.499	HEURI	0.499	GAUSS	0.498	PERMU	0.496
LOOP	0.493	IMPLE	0.492	PROFE	0.491	ALTER	0.490
SUCCE	0.490	HANDL	0.490	QUASI	0.488	DIVIS	0.488
HIGH	0.488	PERIO	0.488	RELEV	0.487	ALLOC	0.486
INVAR	0.486	DIAGN	0.486	SPEED	0.482	POSIT	0.482
INTRO	0.482	CASE	0.482	TERM	0.482	CAPAC	0.482
UTILI	0.480	SELF	0.480	LINKS	0.480	INDEP	0.480
SOLVI	0.479	PAPER	0.477	PRINC	0.475	STEP	0.475
MARKO	0.474	FLOWC	0.473	QUALI	0.472	PRECI	0.471
TOWAR	0.470	CHART	0.470	FAST	0.470	RELAY	0.469
ADDIT	0.469	FUNDA	0.468	STRAT	0.468	ASA	0.468
REQUI	0.466	EXPAN	0.466	DECOD	0.466	FACIL	0.464
DUAL	0.464	360	0.461	NETS	0.459	BOOKS	0.457
EQUIP	0.457	WRITI	0.457	SECON	0.457	DEPEN	0.457
SHIFT	0.457	REALI	0.456	PREDI	0.453	JOURN	0.452
DECCM	0.451	PREPA	0.450	REGIO	0.448	COEFF	0.448
FOURI	0.448	RESOU	0.447	WEIGH	0.445	CONDU	0.445
ACCOU	0.445	CHEBY	0.443	RANK	0.442	PASS	0.442
DOMAI	0.442	VECTO	0.442	RIGHT	0.441	TITLE	0.441
DELAY	0.441	PHYSI	0.441	LIST	0.440	FINDI	0.439
CURRI	0.437	COBOL	0.437	ARTIC	0.436	MOTIO	0.434
PSEUD	0.433	PERSO	0.432	COST	0.432	PLANE	0.432
INQUI	0.432	SCHED	0.431	MAJOR	0.430	SORT	0.430
SUM	0.430	WATER	0.430	JAPAN	0.430	ACADE	0.430
SERVO	0.430	NONPA	0.430	KDF9	0.430	KEYWO	0.430
TUTOR	0.430	ANNOU	0.430	EXECU	0.430	BOARD	0.430

DISTO	0.430	PAGED	0.430	FIXED	0.430	GROWT	0.430
DISCO	0.430	BIBLI	0.429	DIAGR	0.428	SCHOO	0.428
PHRAS	0.428	EXTRE	0.428	SCALE	0.428	SUBRE	0.427
REVIE	0.426	SYMBO	0.424	STIBI	0.423	REMAR	0.422
MEETI	0.421	EXTEN	0.429	LEVEL	0.428	CHOIC	0.419
ELIMI	0.419	PRIOR	0.417	ORTHO	0.417	EDUCA	0.415
LENGT	0.413	SUBMI	0.412	BELL	0.412	BACKW	0.412
PROBS	0.412	COURT	0.412	BORDE	0.412	POLLU	0.412
CEP	0.412	RECEP	0.412	PROMO	0.412	MONOI	0.412
PUFFT	0.412	CALL	0.412	JORDA	0.412	DREDG	0.412
RIGID	0.412	POLES	0.412	ALLIE	0.412	CULTU	0.412
CHOMS	0.412	POLIT	0.412	ERRAT	0.412	BIVAR	0.412
UNSTE	0.412	PREFE	0.412	BOND	0.412	POSTA	0.412
STEPP	0.412	LOCUS	0.412	AERON	0.412	TANK	0.412
BIRTH	0.412	RCA	0.412	SHOCK	0.412	DECOC	0.412
SIMON	0.412	PROLE	0.412	JACKS	0.412	CDS	0.412
APPRA	0.412	RAY	0.412	TRI	0.412	RACHF	0.412
REPLA	0.412	MARK	0.412	HOBBS	0.412	DEMON	0.412
HOT	0.412	RC400	0.412	STEPS	0.412	DISCI	0.412
GAIN	0.412	PREVI	0.412	HELIC	0.412	DOMIN	0.412
FIRM	0.412	PRICI	0.412	SESSI	0.412	REGEL	0.412
TROUB	0.412	DORN	0.412	POSTI	0.412	SLT	0.412
LATEN	0.412	PEACE	0.412	RENAM	0.412	ECMA	0.412
UNSUP	0.412	PAGE	0.412	PLAN	0.412	CANAD	0.412
ORANI	0.412	ONTAR	0.412	SYNAP	0.412	PEEKA	0.412
TRIAN	0.412	ELLIO	0.412	MAGNU	0.412	SIZED	0.412
SUBOP	0.412	PARIT	0.412	TELEC	0.412	ENDED	0.412
SYLLA	0.412	PROCR	0.412	NONCO	0.412	BANDS	0.412
SIZE	0.412	TRACK	0.412	PARTY	0.412	CORNE	0.412
MARKU	0.412	ACOUT	0.412	STAR	0.412	SHEFF	0.412
OCCUR	0.412	ACMCP	0.412	EXCLU	0.412	MEANI	0.412
ORBIT	0.412	AGENT	0.412	BITS	0.412	ROMAN	0.412
PACKE	0.412	PANTA	0.412	OHIO	0.412	PROGA	0.412
MERGI	0.412	ANTIC	0.412	CANCE	0.412	ORDNU	0.412
SCHOL	0.412	FIT	0.412	TRIAL	0.412	I/O	0.412
AMEND	0.412	FCC	0.412	VIROL	0.412	OPTIO	0.412
MERCU	0.412	PANEL	0.412	ATTEM	0.412	DIGES	0.412
CHICA	0.412	SHOP	0.412	TREAT	0.412	LOCI	0.412
COLOU	0.412	CONFR	0.412	LOAD	0.412	CARTO	0.412
APPRE	0.412	UNSTA	0.412	LONDO	0.412	LOSSE	0.412
BROMB	0.412	LANCZ	0.412	APL	0.412	SHELL	0.412
SPATI	0.412	MARKS	0.412	BASSA	0.412	JUMP	0.412
BINDI	0.412	MASTE	0.412	VOYSE	0.412	BAIRS	0.412
VISIO	0.412	MANUS	0.412	WILSC	0.412	ROSEN	0.412
DUPLE	0.412	SIDES	0.412	2314	0.412	KOSHE	0.412
INCOR	0.412	GIER	0.412	LABS	0.412	UNCON	0.412
ATTEN	0.412	LANGE	0.412	UNVOI	0.412	BROWN	0.412
DISAG	0.412	AUTON	0.412	AUDIO	0.412	LAYOU	0.412
TOWER	0.412	HOUSI	0.412	ANCMA	0.412	FIDEL	0.412
RESIS	0.412	HILL	0.412	ROBIN	0.412	KNUTH	0.412
SARDI	0.412	LENSE	0.412	DUALI	0.412	SEMIG	0.412
ROYAL	0.412	STOPP	0.412	SIDE	0.412	GROSS	0.412
FERRI	0.412	NORMS	0.412	SUBSE	0.412	BANKI	0.412
TANDE	0.412	NETHE	0.412	SLOW	0.412	NET	0.412
STAT1	0.412	META	0.412	CLENS	0.412	NONSI	0.412

HAMMI	0.412	BALL	0.412	NONDE	0.412	NONRE	0.412
ICL	0.412	REWRI	0.412	MILNE	0.412	FLUX	0.412
MODAL	0.412	BEREC	0.412	MORPH	0.412	MODES	0.412
IMPRE	0.412	BLIND	0.412	METAB	0.412	MATEM	0.412
DECAD	0.412	GE	0.412	MERGE	0.412	MERSE	0.412
IONIN	0.412	INTEN	0.412	INTEL	0.412	PREVE	0.412
JENKI	0.412	EXPOR	0.412	IRRED	0.412	IONES	0.412
INDIR	0.412	GIRO	0.412	KEIO	0.412	POLYP	0.412
LAGS	0.412	ENZYM	0.412	ISSUE	0.412	JONES	0.412
HARD	0.412	ELEKT	0.412	INACC	0.412	RAPHS	0.412
HIRSC	0.412	DAVIE	0.412	ILL	0.412	ICT	0.412
HADAM	0.412	RETAI	0.412	HIGHE	0.412	REED	0.412
REGIM	0.412	HOLLA	0.412	HOHER	0.412	EASTM	0.412
LONGE	0.412	GIVEN	0.412	PAREN	0.412	ESSO	0.412
FREED	0.412	FIXAT	0.412	RISK	0.412	HASH	0.412
PERT	0.412	METNO	0.412	HIDDE	0.412	HABIT	0.412
HETER	0.412	DIODE	0.412	PENTO	0.412	FRANK	0.412
DOUGL	0.412	ISOMO	0.412	FLOWS	0.412	FRANC	0.412
DEEP	0.412	IMAGE	0.412	FLOWR	0.412	GAAS	0.412
MINER	0.412	OVERD	0.412	GLYCO	0.412	FLORE	0.412
DRUGS	0.412	BIGEL	0.412	EINSC	0.412	LOADI	0.412
EMPHA	0.412	ESTAB	0.412	EDELM	0.412	EXCHA	0.412
BETA	0.412	FABRI	0.412	MAPS	0.412	FLEXI	0.412
FEASI	0.412	ESSAY	0.412	DEFLE	0.412	AREA	0.412
DECID	0.412	KNIGH	0.412	DISSI	0.412	PRACN	0.412
DRIFT	0.412	PURSU	0.412	DATAN	0.412	LAYMA	0.412
DATAF	0.412	LIBER	0.412	DEFER	0.412	FIGUR	0.412
CLARI	0.412	MUSIC	0.412	MOORE	0.412	DEPOS	0.412
SIGNE	0.412	NICHO	0.412	NORM	0.412	DONNE	0.412
CBAC	0.412	MOD	0.412	MAEHL	0.412	DATAM	0.412
LAGRA	0.412	CREDI	0.412	MICHI	0.412	COSMI	0.412
FLOW	0.408	GUIDA	0.407	WORKI	0.407	CAVIT	0.401
STORE	0.401	RACHE	0.401	RESER	0.401	MOVIN	0.401
ASYMM	0.401	KORZH	0.401	GRAHA	0.401	IMPED	0.401
DEMOD	0.401	EVIDE	0.401	TAU	0.401	BRAIN	0.401
CONFO	0.401	CHURC	0.401	BENDI	0.401	BLEND	0.401
AUTO	0.401	AREAS	0.401	UNFOR	0.401	TOLER	0.401
TELEM	0.401	SIMSC	0.401	WESCO	0.401	VERTE	0.401
VIEWP	0.401	STAGE	0.401	SUBCO	0.401	RETRO	0.401
SORTE	0.401	SCALI	0.401	ROTAT	0.401	TREE	0.398
THESA	0.398	DISSE	0.398	MODIF	0.395	IMPUL	0.393
SINGU	0.390	ATLAS	0.389	POSSI	0.389	LAPLA	0.387
TIMET	0.386	FURTH	0.385	ENCOD	0.385	EXPRE	0.384
KUTTA	0.384	SENSE	0.381	EVENT	0.377	REGAR	0.374
POLE	0.374	PEOPL	0.374	DECEN	0.374	FILLI	0.374
NUCLE	0.373	UNIFO	0.372	INSTR	0.366	UNION	0.365
ANSWE	0.362	COMBI	0.362	HARDW	0.361	COMMA	0.361
HEAT	0.360	IFAC	0.360	MEAN	0.360		

TOTAL = 823

APPENDIX D

FINAL INDEX TERM LIST

COMPU	0.994	SYSTE	0.992	PRCCE	0.968	CONTR	0.931
TECHN	0.927	TRANS	0.918	METHO	0.913	GENER	0.909
PROBL	0.900	DATA	0.892	INFOR	0.892	AUTOM	0.882
OPTIM	0.882	FUNCT	0.877	LANGU	0.874	INTER	0.874
LINEA	0.871	ANALY	0.868	STRUC	0.852	PROGR	0.848
ALGOR	0.844	CHEMI	0.844	INDEX	0.836	MULTI	0.833
TIME	0.829	RETRI	0.822	OPERA	0.816	ERROR	0.816
APPLI	0.815	THEOR	0.815	CLASS	0.811	CODES	0.809
MACHI	0.801	SOLUT	0.796	APPRO	0.795	EVALU	0.790
EQUAT	0.786	SEQUE	0.786	DIGIT	0.784	CORRE	0.783
VARIA	0.776	DIFFE	0.775	MATRI	0.774	LITER	0.773
INTEG	0.773	CONVE	0.772	CONST	0.772	SIMUL	0.767
EXPER	0.766	STABI	0.759	AMERI	0.751	DESIG	0.748
RANDO	0.743	NONLI	0.741	SIGNA	0.734	STATE	0.733
RECOG	0.728	REAL	0.722	LINE	0.721	PROBA	0.717
FINIT	0.716	LIBRA	0.714	SCIEN	0.713	NUMBE	0.709
NUMER	0.709	STAND	0.709	STATI	0.706	LOGIC	0.703
ESTIM	0.699	PROPO	0.698	COMMU	0.698	ALGOL	0.698
COMPA	0.698	FORTR	0.696	DYNAM	0.696	GRAPH	0.694
ELEME	0.693	SEARC	0.689	MODEL	0.688	DETER	0.687
RELAT	0.686	MEMOR	0.686	ORDER	0.686	SAMPL	0.683
CALCU	0.682	UNIVE	0.681	MICRO	0.680	PROPE	0.680
REGUL	0.678	BOOLE	0.678	CHANN	0.677	FORMA	0.676
EFFIC	0.675	FILTE	0.675	DISTR	0.675	NOISE	0.674
ORGAN	0.672	ITERA	0.670	MINIM	0.670	COORD	0.669
INPUT	0.669	STORA	0.669	CIRCU	0.666	STOCH	0.663
DISCR	0.663	NOTAT	0.662	FREE	0.662	SERVI	0.661
BINAR	0.657	LARGE	0.654	ALGEB	0.652	CONDI	0.651
NETWO	0.649	RESEA	0.648	CODE	0.645	CERTA	0.644
MAGNE	0.643	FEEDB	0.642	MEDIC	0.640	SUBJE	0.640
SCHEM	0.639	ABSTR	0.638	POLYN	0.637	PARAM	0.637
DIMEN	0.634	DEVEL	0.634	DIREC	0.634	DEFIN	0.633
COMPL	0.633	ELECT	0.632	DOCUM	0.630	FORMU	0.630
SIMPL	0.629	SHARI	0.626	POINT	0.626	CURRE	0.625
SYNTA	0.624	ORDIN	0.622	COMPI	0.621	CONTE	0.621
DISPL	0.620	BOUND	0.618	SINGL	0.618	SYNTH	0.614
INVES	0.614	TABLE	0.613	INVER	0.613	SPECI	0.612
BIT	0.611	IBM	0.611	EFFEC	0.611	LIMIT	0.611
CHARA	0.610	MANAG	0.610	REDUC	0.607	PRODU	0.605
ARITH	0.605	CONTI	0.604	TAPE	0.601	SERIA	0.600
DERIV	0.600	ADAPT	0.599	VALUE	0.598	FORM	0.598
STUDY	0.598	PERFO	0.596	PLATE	0.595	DECIS	0.595
SELEC	0.593	MECHA	0.592	ORIEN	0.591	PATTE	0.591
HYBRI	0.590	MODUL	0.590	PRINT	0.590	BUSIN	0.590
SOURC	0.589	TESTI	0.589	OUTPU	0.588	FREQU	0.588
EIGEN	0.587	PHASE	0.583	SWITC	0.582	PRACT	0.581
SOCIA	0.580	ECONO	0.578	ENGIN	0.578	BASED	0.576
FILE	0.576	ASYMP	0.574	ABSOL	0.572	COLLE	0.571

SQUAR	0.571	ACCES	0.570	ACTIV	0.569	PRESE	0.568
DESCR	0.567	REPRE	0.567	SPECT	0.567	COMPO	0.566
TEST	0.565	CENTR	0.563	NORMA	0.563	THRES	0.562
MATHE	0.561	FACTO	0.560	PATEN	0.560	SERIE	0.559
BLOCK	0.557	CATAL	0.557	SMALL	0.555	ASPEC	0.552
USER	0.551	CRITE	0.551	IMPRO	0.550	SOFTW	0.548
SPACE	0.548	MANIP	0.547	AMPLI	0.545	IDENT	0.545
EQUIV	0.545	TYPE	0.544	GRAMM	0.543	INDUS	0.543
NON	0.542	CYCLI	0.542	NOTE	0.542	SYNCH	0.542
SYMPO	0.540	SEPAR	0.537	REPOR	0.537	CONCE	0.537
RAPID	0.536	ANALO	0.534	SYMME	0.534	CONFE	0.534
GROUP	0.533	RUNGE	0.532	RESUL	0.532	FIELD	0.529
CENTE	0.529	ALPHA	0.528	BASE	0.527	DEPAR	0.526
FILM	0.526	HARMO	0.526	PARSE	0.526	RECTA	0.526
STUDI	0.525	INSUR	0.525	BRIEF	0.525	KONVE	0.525
COLLA	0.525	PREFI	0.525	ARGUM	0.525	PL1	0.525
REDUN	0.525	FREDH	0.525	FACT	0.525	DEVIA	0.525
NATIO	0.524	DEVIC	0.524	SURVE	0.523	LAW	0.520
TERMI	0.520	CARDS	0.520	CARD	0.520	REFER	0.520
QUADR	0.519	VIEW	0.518	PUNCH	0.517	COMME	0.517
PARAL	0.517	ADMIN	0.514	CITAT	0.513	TEACH	0.511
DETEC	0.510	SET	0.509	PAPER	0.509	MONIT	0.508
PURPO	0.507	AWARE	0.507	QUANT	0.504	REMOT	0.504
ONLIN	0.501	MARKE	0.500	PLANN	0.495	PACKA	0.494
COMPR	0.493	PHYSI	0.491	PIECE	0.491	LOCAL	0.490
RESPO	0.489	EXTRA	0.489	ASSIG	0.488	QUEUE	0.488
HAND	0.486	ASSOC	0.485	SETS	0.485	PERIO	0.484
SENSI	0.484	CODIN	0.484	REGIS	0.480	PUBLI	0.480
CONSI	0.480	RECUR	0.477	GAMES	0.476	HEURI	0.475
ROOTS	0.475	NATUR	0.474	ASLIB	0.473	LOOP	0.472
PULSE	0.471	GAUSS	0.470	HANDL	0.469	SOLVI	0.468
PICTU	0.468	EDUCA	0.468	SPEED	0.467	SECON	0.466
PERMU	0.465	PRINC	0.462	HIGH	0.461	DIVIS	0.460
CAPAC	0.460	SELF	0.460	QUASI	0.460	IMPLE	0.459
ALTER	0.459	DIAGN	0.459	SUCCE	0.459	PROFE	0.458
ALLOC	0.457	TERM	0.457	RELEV	0.456	INVAR	0.454
FACIL	0.453	CASE	0.451	POSIT	0.450	INTRO	0.449
STEP	0.449	INDEP	0.449	UTILI	0.449	LINKS	0.448
COEFF	0.447	FOURI	0.447	FAST	0.447	QUALI	0.444
MARKO	0.444	PRECI	0.443	DECOD	0.442	FLOWC	0.441
STRAT	0.441	TOWAR	0.440	RELAY	0.440	CHART	0.440
ADDIT	0.439	FUNDA	0.439	ASA	0.438	EXPAN	0.438
REQUI	0.438	PREDI	0.433	DUAL	0.433	REALI	0.433
EQUIP	0.432	BOOKS	0.432	WRITI	0.432	360	0.430
JOURN	0.430	NETS	0.429	DEPEN	0.427	SHIFT	0.427
DECCM	0.424	RESOU	0.423	PREPA	0.421	REGIO	0.420
CHEBY	0.419	WEIGH	0.418	CONDU	0.418	TITLE	0.416
VECTO	0.415	DELAY	0.414	ACCOU	0.414	RANK	0.412
PSEUD	0.412	RIGHT	0.412	PASS	0.412	DOMAI	0.412
FINDI	0.411	CURRI	0.410	LIST	0.410	REVIE	0.409
ARTIC	0.408	COBOL	0.408	BIBLI	0.406	EXTEN	0.404
PERSO	0.404	LEVEL	0.404	SCHOO	0.404	COST	0.404
DIAGR	0.404	MOTIO	0.404	PLANE	0.403	INQUI	0.403
SCHED	0.402	PAGED	0.402	KDF9	0.402	NONPA	0.402
DISCO	0.402	GROWT	0.402	KEYWO	0.402	BOARD	0.402

MAJOR	0.402	ACADE	0.402	JAPAN	0.402	SERVO	0.402
SUM	0.402	SORT	0.402	DISTO	0.402	TUTOR	0.402
FIXED	0.402	WATER	0.402	EXECU	0.402	ANNOU	0.402
EXTRE	0.401	PHRAS	0.401	SCALE	0.400	SUBRE	0.399
SYMBO	0.398	STIBI	0.394	REMAR	0.393	MEETI	0.393
ORTHO	0.391	CHOIC	0.391	ELIMI	0.391	PRIOR	0.390
LENGT	0.388	BACKW	0.384	SHOCK	0.384	DOUGL	0.384
STEPS	0.384	EXCLU	0.384	BELL	0.384	LOCUS	0.384
ROBIN	0.384	TANK	0.384	TRI	0.384	JORDA	0.384
AGENT	0.384	PARIT	0.384	ALLIE	0.384	DORN	0.384
SYNAP	0.384	CHOMS	0.384	MONOI	0.384	METNO	0.384
JACKS	0.384	POSTA	0.384	BOND	0.384	MARKU	0.384
TELEC	0.384	ATTEN	0.384	SIDE	0.384	MARK	0.384
MERCU	0.384	NONDE	0.384	POSTI	0.384	DIGES	0.384
HOBBS	0.384	UNSTE	0.384	CARTO	0.384	LAGS	0.384
ANOMA	0.384	GAAS	0.384	APL	0.384	DUPLE	0.384
AERON	0.384	RETAI	0.384	STAT1	0.384	ELLIO	0.384
BINDI	0.384	ONTAR	0.384	ACOUT	0.384	DUALI	0.384
COLOU	0.384	STEPP	0.384	PLAN	0.384	DECAD	0.384
GIER	0.384	LOCI	0.384	SEMIG	0.384	ICL	0.384
SPATI	0.384	LIBER	0.384	TOWER	0.384	MERGI	0.384
TRACK	0.384	NORMS	0.384	CHICA	0.384	PREVI	0.384
SCHOL	0.384	RIGID	0.384	HELIC	0.384	LANCZ	0.384
SESSI	0.384	PERT	0.384	ERRAT	0.384	I/O	0.384
PANTA	0.384	GIRO	0.384	VISIO	0.384	DISAG	0.384
RENAM	0.384	ORDNU	0.384	VOYSE	0.384	FIDEL	0.384
AUDIO	0.384	CALL	0.384	SARDI	0.384	META	0.384
2314	0.384	HOHER	0.384	ANTIC	0.384	MINER	0.384
ROSEN	0.384	BAIRS	0.384	NONCO	0.384	IMPRE	0.384
STOPP	0.384	MERGE	0.384	UNCON	0.384	BIRTH	0.384
FERRI	0.384	NET	0.384	TANDE	0.384	EXPOR	0.384
LENSE	0.384	FLOWS	0.384	SLOW	0.384	SUBMI	0.384
BROWN	0.384	EXCHA	0.384	DREDG	0.384	ECMA	0.384
RESIS	0.384	MODAL	0.384	UNVOI	0.384	RECEP	0.384
MAGNU	0.384	POLYP	0.384	BASSA	0.384	STAR	0.384
SUBSE	0.384	RC400	0.384	AMEND	0.384	APPRA	0.384
WILSO	0.384	HETER	0.384	ACMCP	0.384	BEREC	0.384
KNUTH	0.384	SUBOP	0.384	IONES	0.384	IMAGE	0.384
ILL	0.384	REPLA	0.384	DEFLE	0.384	MICHI	0.384
LABS	0.384	LATEN	0.384	REED	0.384	RISK	0.384
PROMO	0.384	SIMON	0.384	MERSE	0.384	NORM	0.384
LOAD	0.384	BORDE	0.384	OHIO	0.384	BANKI	0.384
ISSUE	0.384	SLT	0.384	POLES	0.384	DECOC	0.384
MASTE	0.384	SHEFF	0.384	MOD	0.384	SIDES	0.384
NONSI	0.384	FIRM	0.384	ESSAY	0.384	BITS	0.384
PRICI	0.384	HOT	0.384	CREDI	0.384	PURSU	0.384
DECID	0.384	TROUB	0.384	PEACE	0.384	OVERD	0.384
PROLE	0.384	TREAT	0.384	NICHO	0.384	BETA	0.384
METAB	0.384	GAIN	0.384	PANEL	0.384	HIDDE	0.384
POLLU	0.384	SHOP	0.384	RACHF	0.384	CANAD	0.384
DATAN	0.384	ORANI	0.384	FRANK	0.384	GROSS	0.384
POLIT	0.384	TRIAN	0.384	MAPS	0.384	REWRI	0.384
LAYOU	0.384	UNSUP	0.384	DONNE	0.384	BIVAR	0.384
OPTIO	0.384	ROYAL	0.384	EDELM	0.384	LAGRA	0.384
FABRI	0.384	SIZE	0.384	INACC	0.384	PREVE	0.384

ORBIT	0.384	CORNE	0.384	LAYMA	0.384	SIGNE	0.384
LONDO	0.384	SYLLA	0.384	PROBS	0.384	REGIM	0.384
PAGE	0.384	INTEN	0.384	MODES	0.384	CDS	0.384
ESTAB	0.384	ROMAN	0.384	MARKS	0.384	INCOR	0.384
PROCR	0.384	PROGA	0.384	FLUX	0.384	ELEKT	0.384
HASH	0.384	MEANI	0.384	JONES	0.384	PRACN	0.384
DRIFT	0.384	LONGE	0.384	MATEM	0.384	AUTON	0.384
NETHE	0.384	VIROL	0.384	DRUGS	0.384	FCC	0.384
DIODE	0.384	ATTEM	0.384	HAMMI	0.384	SHELL	0.384
FLOWR	0.384	TRIAL	0.384	FIGUR	0.384	BANDS	0.384
DEFER	0.384	BIGEL	0.384	DEPOS	0.384	APPRE	0.384
MORPH	0.384	UNSTA	0.384	OCCUR	0.384	PAREN	0.384
DEEP	0.384	BROMB	0.384	JUMP	0.384	PENTO	0.384
HIGHE	0.384	CONFR	0.384	NONRE	0.384	MAEHL	0.384
FEASI	0.384	CLARI	0.384	LOSSE	0.384	BLIND	0.384
LANGE	0.384	COURT	0.384	RCA	0.384	CLENS	0.384
FIXAT	0.384	DOMIN	0.384	HOUSI	0.384	MOORE	0.384
IRRED	0.384	RAPHS	0.384	COSMI	0.384	AREA	0.384
EASTM	0,384	ISOMO	0.384	FRANC	0.384	CANCE	0.384
MANUS	0.384	GE	0.384	KEIO	0.384	SIZED	0.384
ESSO	0.384	BALL	0.384	CULTU	0.384	FREED	0.384
DISSI	0.384	MUSIC	0.384	PARTY	0.384	CBAC	0.384
HOLLA	0.384	ENZYM	0.384	PUFFT	0.384	CEP	0.384
RAY	0.384	LOADI	0.384	GIVEN	0.384	DEMON	0.384
INDIR	0.384	HILL	0.384	DISCI	0.384	DAVIE	0.384
DATAF	0.384	KNIGH	0.384	FLORE	0.384	IONIN	0.384
ICT	0.384	PREFE	0.384	MILNE	0.384	JENKI	0.384
ENDED	0.384	FLEXI	0.384	REGEL	0.384	GLYCO	0.384
PEEKA	0.384	PACKE	0.384	KOSHE	0.384	HADAM	0.384
FIT	0.384	EINSC	0.384	INTEL	0.384	HARD	0,384
DATAM	0.384	HABIT	0.384	EMPHA	0.384	HIRSC	0.384
FLOW	0.381	GUIDA	0.380	WORKI	0.380	MODIF	0.379
SINGU	0.375	IMPUL	0.374	STAGE	0.373	CAVIT	0.373
SORTE	0.373	STORE	0.373	SUBCO	0.373	RESER	0.373
ROTAT	0.373	SIMSC	0.373	RETRO	0.373	ASYMM	0.373
VIEWP	0.373	GRAHA	0.373	SCALI	0.373	TOLER	0.373
EVIDE	0.373	TAU	0.373	VERTE	0.373	MOVIN	0.373
AREAS	0.373	BENDI	0.373	WESCO	0.373	CONFO	0.373
UNFOR	0.373	TELEM	0.373	DEMOD	0.373	IMPED	0.373
RACHE	0.373	KORZH	0.373	AUTO	0.373	BLEND	0.373
BRAIN	0.373	CHURC	0.373	DISSE	0.372	THESA	0.372
TREE	0.372	LAPLA	0.365	FURTH	0.364	ENCOD	0.364
POSSI	0.362	ATLAS	0.362	TIMET	0.361	KUTTA	0.360
EXPRE	0.360	SENSE	0.357	EVENT	0.351	REGAR	0.349
NUCLE	0.349	PEOPL	0.349	DECEN	0.349	FILLI	0.349
POLE	0.349	UNIFC	0.347	INSTR	0.344		

TOTAL = 815

APPENDIX E

FURTHER SEARCH EXAMPLES

(I) Multi-parameter Search Example One

```
QUE                                                      60  70
    MULTI-PARAMETER SEARCH EXAMPLE ONE
AND T MANAGEMENT                                            300
 OR T INFORMATION                                           100
 OR T SYSTEMS                                               100
AND T BUSINESS                                              300
 OR T INFORMATION                                           100
 OR T SYSTEMS                                               100
END
```

The value of T' and T are respectively 0.466 and 0.733. After normalization, the first request vector becomes (management, information, systems) which has the corresponding set of weights (0.905, 0.302, 0.302). The set of search output for parameter one is given as follows:

Relevance Value	Document
0.655	AMDOA70210204MATHE/USING TIP SYSTE ASSIS FILE MANAG EXERC
0.598	AMDOA70210209HIGGE/MAYS /SALUT ASIS MANAG SYSTE EXERC USING PL1 USING GENER PURPO SYSTE

0.505	AMDOA69200111HELMK/MANAG COST ACCOU TECHN INFOR CENTE
0.502	AMDOA70210163COCKR/SMITH/DOUGL/BENDE/APPLI MANAG COST ACCOU SCIEN INFOR
0.477	AMDOA70210214OLLE /GAGNO/SOLUT ASIS FILE MANAG EXERC USING KCA UL1
0.472	AMDOA70210219BLOOM/APPLI CAPRI ASIS FILE MANAG EXERC

Since the highest relevance value of this set of documents is 0.655 which is in the interval [0.466, 0.733], iterative search is required. The new query vector is (management, information, systems, using, tip, assistance, file, exercise) which has the corresponding set of weights (0.597, 0.302, 0.802, 0.0, 0.0, 0.0, 0.576, 0.0). Note that the new request vector for parameter one now includes FILE as a significant term. The value of α_1 is 0.504. By using a cutoff value of 0.550, the final set of search output for parameter one is given as follows:

Relevance Value	Document
0.963	AMDOA70210204MATHE/USING TIP SYSTE ASSIS FILE MANAG EXERC
0.852	AMDOA70210209HIGGE/MAYS /SALUT ASIS MANAG SYSTE EXERC USING PL1 USING GENER PURPO SYSTE
0.699	AMDOA66170026PARKE/USERS PLACE INFOR SYSTE
0.670	AMDOA70210040HOLBR/THREA FILE RETRI SYSTE
0.631	AMDOA70210274BURCH/ROLE FEDER GOVER INFOR SYSTE EDUCA
0.572	AMDOA69200279SWANS/USER ORIEN INFOR SYSTE
0.566	ACJ 69010201AUSTI/HOLDE/RECEN DEVEL DAD SYSTE

0.565	AMDOA68190181WALL /POSSI ARTIC INFOR SYSTE NETWO
0.550	AMDOA68190221JORDA/FRAME COMPA SDI SYSTE
0.550	AMDOA70210160RICHM/COMPA SYSTE LABOR

After normalization, the second request vector becomes (business, information, systems) which has the corresponding set of weights (0.905, 0.302, 0.302). The set of search output for parameter two is given as follows:

Relevance Value	Document
0.674	AMDOA68190265CUETO/USING INFOR LIFE INSUR BUSIN WORLD

Since the only relevance value is 0.674 which is in the interval [0.466, 0.733], iterative search is required. The new query vector is (business, information, systems, using, life, insurance, world) which has the corresponding set of weights (0.571, 0.806, 0.302, 0.0, 0.0, 0.525, 0.0). Note that the new request vector for parameter two now includes INSURANCE as a significant term. The value of α_1 is 0.565. By using a cutoff value of 0.550, the final set of search output for parameter two is given as follows:

Relevance Value	Document
0.965	AMDOA68190265CUETO/USING INFOR LIFE INSUR BUSIN WORLD
0.696	AMDOA68190286MILLE/PSYCH INFOR

0.696	AMDOA70210089OTTEN/DEBON/METAS INFOR
0.659	AMDOA66170026PARKE/USERS PLACE INFOR SYSTE
0.639	AMDOA70210004HUMPH/INFOR PEACE
0.622	AMDOA70210274BURCH/ROLE FEDER GOVER INFOR SYSTE EDUCA
0.604	AMDOA68190375COTTR/EVALU COMPR SCIEN TECHN INFOR NUCLE SAFET INFOR CENTE
0.600	AMDOA65160291GARVI/INFOR SURVE MODER LINGU
0.597	AMDOA67180235BUCHA/HUTTO/ANALY AUTOM HANDL TECHN INFOR NUCLE SAFET INFOR
0.596	AMDOA68190200OCONN/QUEST CONCE INFOR NEED
0.584	AMDOA70210095BROMB/ECONO INFOR
0.563	AMDOA69200279SWANS/USER ORIEN INFOR SYSTE
0.558	AMDOA69200039LUNIN/ACADE INFOR CENTE
0.556	AMDOA68190181WALL /POSSI ARTIC INFOR SYSTE NETWO
0.556	AMDOA68190305THOMP/ORGAN INFOR

(II) Multi-parameter Search Example Two

```
QUE                                                        60  70
    MULTI-PARAMETER SEARCH EXAMPLE TWO
AND T COMPUTER                                                300
 OR T SIMULATION                                              500
 OR T SMALL                                                   100
 OR T INFORMATION                                             300
 OR T SYSTEM                                                  200
AND T COMPUTER                                                300
 OR T SIMULATION                                              500
 OR T SMALL                                                   100
 OR T INFORMATION                                             300
 OR T NETWORK                                                 200
END
```

After normalization, the first request vector becomes (computer, simulation, small, information, system) which has the corresponding set of weights (0.433, 0.722, 0.144, 0.433, 0.289). The value of T' and T are respectively 0.466 and 0.733. The set of search output for parameter one is given as follows:

Relevance Value	Document
0.907	AMDOA68190120CARAS/COMPU SIMUL SMALL INFOR SYSTE
0.625	AMDOA68190363BAKER/NANCE/USE SIMUL STUDY INFOR STORA RETRI SYSTE

0.552	AMDOA70210285JERMA/PROMI DEVEL COMPU ASSIS INFOR
0.534	AMDOA64150142BOURN/FORD /COST ANALY SIMUL PROCE EVALU LARGE INFOR SYSTE
0.504	AMDOA66170026PARKE/USERS PLACE INFOR SYSTE
0.489	AMDOA68190278MCCON/COMPU GRAPH ASSEM LINE INFOR
0.476	AMDOA70210274BURCH/ROLE FEDER GOVER INFOR SYSTE EDUCA

After normalization, the second request vector becomes (computer, simulation, small, information, network) which has the corresponding set of weights (0.433, 0.722, 0.144, 0.433, 0.289). The set of search output for parameter two is given as follows:

Relevance Value	Document
0.758	AMDOA68190120CARAS/COMPU SIMUL SMALL INFOR SYSTE
0.552	AMDOA70210285JERMA/PROMI DEVEL COMPU ASSIS INFOR
0.489	AMDOA68190278MCCON/COMPU GRAPH ASSEM LINE INFOR
0.479	AMDOA68190363BAKER/NANCE/USE SIMUL STUDY INFOR STORA RETRI SYSTE

Since the highest relevance values of the two sets of documents are greater than 0.733, this search request does not require iterative searches. The final set of search output to be presented to the user in both examples will be the set of documents common to the two sets of output in respond to the two parameters.

B30030